Exploring Data Efficiency of Foundation Models for Fine-Grained Image Classification of Lepidopteran Species

Bachelor Thesis

Bachelor of Science in Applied Computer Science

Leopold Gierz

March 17, 2025

Supervisor:

1st: Prof. Dr. Christian Ledig
2nd: Jonas Alle, M.Sc.

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

Abstract

Citizen science initiatives have become a cornerstone in advancing scientific research and democratizing access to knowledge. One of the most impactful ways this is achieved is through large-scale data collection, which is crucial for biodiversity studies. This work utilizes a large dataset of 1.39 million images of butterflies and moths (belonging to the order of Lepidoptera), sourced from Observation.org. The dataset was split into nine variants to examine how different amounts of samples and species labels affect the performance of foundation models combined with classifiers for fine-grained image classification.

The feature embeddings extracted from DINOv2-Giant and ResNet-50 were used to train and evaluate classifiers such as Logistic Regression and K-Nearest Neighbors (KNN) across the dataset variants. Cross-validation was employed to ensure robust performance estimation. The results showed that DINOv2 consistently outperformed ResNet-50, achieving a top-1 accuracy of 92.78% when paired with Logistic Regression on the optimal dataset variant (277 species, 3,000 samples each). In contrast, ResNet-50 reached only 70.34% under similar conditions. Notably, DINOv2 demonstrated strong generalization capabilities without requiring fine-tuning, whereas ResNet-50 may benefit from additional tuning to achieve comparable results.

Despite concerns about the quality and variability of citizen science data, both foundation models exhibited resilience to variations in sample size and species count. For instance, with a dataset containing 589 species and only 500 samples each, DINOv2 still achieved 86.77% top-1 accuracy, while ResNet-50 reached 57.81%. Logistic Regression proved more effective than KNN, likely due to its ability to establish global decision boundaries, which seem to be advantageous in this context.

This research highlights the potential of leveraging citizen science data with foundation models to deliver impactful results in ecological monitoring and conservation efforts. The findings suggest that even with limited data, fine-grained image classification can be performed with relatively high precision.

Abstract

Citizen-Science-Initiativen sind zu einem Eckpfeiler der Förderung wissenschaftlicher Forschung und der Demokratisierung des Zugangs zu Wissen geworden. Eine der wirkungsvollsten Möglichkeiten, dies zu erreichen, sind groß angelegte Datenerhebungen, die für Studien zur biologischen Vielfalt entscheidend sind. In dieser Arbeit wird ein großer Datensatz von 1,39 Millionen Bildern von Schmetterlingen und Motten (aus der Ordnung der Lepidoptera) verwendet, der von der Citizen-Science-Plattform Observation.org stammt. Der Datensatz wurde in neun Varianten aufgeteilt, um zu untersuchen, wie sich unterschiedliche Mengen an Daten auf die Leistung von Basis-Modellen (en: foundation models) in Kombination mit Klassifikatoren (en: classifier) für die feinkörnige Bildklassifikation (en: fine-grained image classification) auswirken.

Die aus DINOv2-Giant und ResNet-50 extrahierten Merkmale (en: feature embeddings) wurden verwendet, um Klassifikatoren wie die logistische Regression und K-nächste Nachbarn (KNN) für die verschiedenen Varianten des Datensatzes zu trainieren und zu bewerten. Die Kreuzvalidierung (en: cross-validation) wurde eingesetzt, um eine robuste Leistungseinschätzung zu gewährleisten. Die Ergebnisse zeigten, dass DINOv2 ResNet-50 durchgängig übertraf und eine Top-1-Genauigkeit von 92,78% erreichte, wenn es mit der logistischen Regression auf der optimalen Datensatzkonfiguration (277 Arten, je 3.000 Bilder) kombiniert wurde. Im Gegensatz dazu erreichte ResNet-50 unter ähnlichen Bedingungen nur 70,34%. Bemerkenswert ist, dass DINOv2 starke Verallgemeinerungsfähigkeiten zeigte, ohne dass eine Feinabstimmung erforderlich war, während ResNet-50 von einer zusätzlichen Abstimmung profitieren könnte, um vergleichbare Ergebnisse zu erzielen.

Trotz der Bedenken hinsichtlich der Qualität und Variabilität von Citizen-Science-Daten zeigten beide Basis-Modelle eine hohe Widerstandsfähigkeit gegenüber Schwankungen bei der Bildermenge und der Artenzahl. Bei einem Datensatz mit 589 Arten und jeweils nur 500 Bildern erreichte DINOv2 beispielsweise eine Top-1-Genauigkeit von 86,77%, während ResNet-50 57,81% erreichte. Die logistische Regression erwies sich als effektiver als KNN, was wahrscheinlich auf ihre Fähigkeit zurückzuführen ist, globale Entscheidungsgrenzen festzulegen, die in diesem Zusammenhang von Vorteil zu sein scheinen.

Diese Untersuchung unterstreicht das Potenzial der Nutzung von Citizen-Science-Daten mit Basis-Modellen, um wirkungsvolle Ergebnisse bei der ökologischen Überwachung und bei Naturschutzmaßnahmen zu erzielen. Die Ergebnisse deuten darauf hin, dass selbst mit begrenzten Daten eine feinkörnige Bildklassifizierung mit relativ hoher Präzision durchgeführt werden kann.

Acknowledgements

Hereby I want to thank Jonas Alle, my supervisor, to whom I have to thank for this interesting and applied topic. Furthermore he was constantly encouraging to follow new pathways by pitching fresh ideas and providing new perspectives – or explaining some details here and there. The good humor he brought to the meetings was very motivating!

The code with which the experiments were conducted, is provided on my personal GitHub and accessible under the following link:

https://github.com/leo-grz/BachelorThesis_LepidopteraFGIC.git.

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
KNN	K-Nearest Neighbor
LR	Logistic Regression
CV	Cross-validation
PCA	Principal Component Analysis
UMAP	Uniform Manifold Approximation and Projection
FM	Foundation Model
CLF	Classifier
GBIF	Global Biodiversity Information Facility

1 Introduction

Lepidoptera, comprising butterflies and moths, are one of the most diverse and ecologically significant insect orders and are playing crucial roles as pollinators, herbivores, and prey for various species. (Hajizadeh et al., 2022) With an estimated species amount of half million, Lepidoptera represent the largest radiation of primarily herbivorous organisms and account for a substantial portion of insect biodiversity. (Kristensen et al., 2007) Their ecological importance extends beyond their sheer numbers, as they serve as reliable bioindicators of ecosystem health and anthropogenic landscape transformations. (Vila, 2025)

In recent years, citizen science has emerged as a powerful tool for biodiversity research, particularly in the study of Lepidoptera. The widespread interest in butterflies and moths among the general public has led to the creation of numerous citizen science projects aimed at collecting valuable data on these species. These initiatives not only contribute to scientific knowledge but often also foster public engagement in conservation efforts. (Lewandowski and Oberhauser, 2016)

However, the influx of citizen-generated data presents both opportunities and challenges for researchers. While citizen science projects provide access to vast amounts of information, the unstructured nature of this data can lead to biases and inconsistencies. (Cordis, 2023) This highlights the critical need for efficient data processing and classification systems in biodiversity research.

The work addresses this challenge by investigating the performance of different foundation models combined with common classifiers on various combinations of label and sample amounts from a large Lepidoptera image dataset. This research seeks to clarify the data requirements essential for precise species classification, offering valuable insights to refine the data requirements for projects using citizen science data.

By focusing on data efficiency, this study could enable citizen science projects to streamline their data collection efforts, concentrating on the most informative types and quantities of data. This work tries to ultimately contribute to a more effective biodiversity monitoring and enhanced conservation strategies.

2 Related Work

Recent advances in deep learning have revolutionized fine-grained visual classification, leading to substantial improvements in biodiversity monitoring and insect studies. The two studies presented below serve as key references, as each tackles the challenges of small inter-class differences and significant intra-class variations in insect classification by employing different architectures and datasets.

Pucci et al. evaluated nine deep learning models — including CNNs, vision transformers (ViTs), and locality-based vision transformers (LBVTs) — on heavily imbalanced insect image datasets sourced from the citizen science initiative Observation.org: One dataset, containing observations of the order of Odonata (containing 628,189 samples over 235 species) and another of the order Coleoptera (comprising 849,296 samples over 3,087 species). Their results revealed that while ViTs offered rapid inference and computational efficiency, LBVTs achieved the best overall classification performance and feature embedding quality, with the top LBVT model reaching a top-1 accuracy of 89.8% or 93.6%, depending on the dataset. LBVTs are hybrid models that merge the local feature extraction strengths of CNNs (by the use of architectures like InceptionV3, EfficientNetV2, and ResNet-50) with the global context awareness of transformers. Their experiments also encompassed testing a ResNet-50, achieving a top-1 accuracy of 86.7% and 90.80%. All results were derived using fine tuned models. (Pucci et al., 2024)

Chang et al. focused on the fine-grained classification of butterflies and moths using a manually annotated dataset of 14,270 samples over 636 species — derived from various sources, such as Wikipedia, Bing API and Google Search. They fine-tuned several classical deep convolutional neural networks, including VGG-19, Inception-v3, ResNet18, and ResNet34, on their dataset to adapt these pre-trained models to the nuances of butterfly and moth imagery. Their experiments revealed that these CNN architectures can deliver robust performance when provided with high-quality, well-prepared data. In fact, the best performing model achieved an accuracy of 71.5% on a ResNet-18. (Chang et al., 2017)

While this experiment relates to the mentioned studies as examples of successful fine-grained insect classification using foundation models, it differs in its approach by not directly fine-tuning them. Instead, it employs different classifiers which are trained on feature embeddings extracted of different foundation models, an approach commonly referred to as **probing**. Within this setup, this work focuses on evaluating data efficiency on **balanced datasets**, analyzing the impact of data quantities on the performance in the context of highly variable data quality, a common challenge in citizen science projects.

3 Theoretical Background

In order to understand the experiment's structure and its working principles, some of the core components will be elaborated on in greater detail in order to enhance interpretability of the results and provide a deeper understanding of the underlying methodology.

3.1 Foundation Models

Foundation models are large-scale models trained on extensive datasets to learn general-purpose representations, often used for feature extraction. Their embeddings can be leveraged by simpler classifiers, such as **K-Nearest Neighbors (KNN)** or **Logistic Regression (LR)**, for downstream tasks. **ResNet**, a convolutional neural network (CNN), excels at extracting localized spatial features through its residual learning mechanism. In contrast, **DINO**, a vision transformer (ViT)-based model, uses self-supervised learning to capture global contextual relationships in images. These architectural differences make ResNet and DINO complementary tools for generating feature embeddings.

3.1.1 ResNet

ResNet (Residual Neural Network) is a groundbreaking deep learning architecture introduced by He at al. in their paper, *"Deep Residual Learning for Image Recognition"* which also won first places across several categories in the ILSVRC 2015 competition. He et al. (2016) The architecture was developed to address the degradation problem observed in very deep neural networks, where increasing network depth leads to higher training error, contrary to expectations. This problem hindered the performance of deep networks, despite their potential for learning more complex features. ResNet revolutionized deep learning by introducing the concept of *residual learning*, enabling the training of networks with hundreds of layers.

Characteristics The key innovation of ResNet is its use of *residual blocks*, which introduce skip connections (also called shortcut connections). These skip connections allow the network to bypass one or more layers, enabling it to learn the difference (or residual) between the input and the desired output, rather than learning the output directly. Mathematically, if the desired transformation is $H(x)$, the residual block learns $F(x) = H(x) - x$, where x is the input. This residual learning approach makes it easier for the network to optimize very deep architectures.

By using skip connections, ResNet addresses two major challenges in deep learning:

- **Vanishing Gradients**: Skip connections allow gradients to flow directly through the network, preventing them from becoming too small during backpropagation. This makes it possible to train networks with hundreds of layers.

- **Degradation Problem**: In very deep networks, adding more layers can sometimes lead to higher training error. Residual blocks mitigate this issue by allowing the network to learn small adjustments (residuals) to the input, rather than forcing it to learn complex transformations from scratch.

These innovations make ResNet highly efficient and capable of achieving state-of-the-art performance with fewer parameters compared to earlier architectures like VGG and AlexNet. (He et al., 2016)

Training Data: ImageNet1k ResNet was originally trained on the `ImageNet1k` dataset, a large-scale benchmark for image classification. ImageNet1k contains 1.2 million training images across 1,000 classes, making it a robust dataset for evaluating deep learning models. In the experiment, the `ImageNet1kV2` weights was used. This means, that it was originally trained on ImageNet1k but fine-tuned to perform well on both, the ImageNet1k validation set and ImageNet1kV2, which is a newer dataset designed to mitigate dataset biases and improve generalization. (Recht et al., 2019) ImageNet1kV2 includes additional images collected using the same class labels as ImageNet1k but with a more rigorous sampling process. It is designed to reduce overfitting to the original ImageNet1k validation set, providing a more robust evaluation benchmark.

Variants of ResNet ResNets have been developed in various depths to address different computational needs and application scenarios. The original ResNet paper introduced models with 18, 34, 50, 101, and 152 layers, referred to as `ResNet-18`, `ResNet-34`, `ResNet-50`, `ResNet-101`, and `ResNet-152`, respectively. Shallower networks, such as `ResNet-18` and `ResNet-34`, are computationally efficient and suitable for tasks with limited resources. Deeper networks, including `ResNet-50`, `ResNet-101`, and `ResNet-152`, offer increased representational capacity, making them effective for complex tasks. **ResNet-50**, in particular, is noted for its balance between depth and computational load, making it a popular choice for various applications.

Compared to earlier architectures like VGG and AlexNet, ResNet achieves superior performance with fewer parameters and computational requirements. For example, ResNet-50 outperforms VGG-16 on ImageNet1k while being computationally more efficient. (He et al., 2016)

Application to Fine-Grained Image Classification ResNet-50 was chosen for the experiments as its widespread use across various domains ensures that results from experiments using ResNet-50 are highly comparable with other studies. This comparability might be particularly valuable when evaluating the impact of different data amounts on model performances.

3.1.2 DINOv2

DINOv2 is a state-of-the-art self-supervised vision model introduced by Meta AI in 2023. Building on the success of its predecessor, DINO, DINOv2 leverages the vision transformer (ViT) architecture and a self-supervised learning framework to learn powerful visual representations without requiring labeled data. Unlike traditional supervised models, DINOv2 uses a teacher-student distillation approach, where the model learns by matching predictions of different augmented views of the same image. This allows DINOv2 to capture both local and global features, making it highly effective for tasks like fine-grained image classification. (Oquab et al., 2024)

Characteristics DINOv2's self-supervised training in combination with the `LVD-142M` dataset enables it to generate high-quality feature embeddings that generalize well across diverse datasets. The model furthermore excels at capturing details and long-range dependencies in images. This makes it particularly suitable for tasks where subtle differences between classes, such as distinguishing between insect species, are critical. Additionally, DINOv2's ability to learn from unlabeled data makes it highly scalable and adaptable to new domains. (Oquab et al., 2024)

Training Data: LVD-142M The performance of a model is largely influenced by the size and quality of the dataset used during training. In the development of DINOv2, researchers assembled a curated dataset (referred to as `LVD-142M`) consisting of 142 million images. This dataset was assembled through an automated pipeline designed to select diverse and representative images from a vast pool of approximately 1.2 billion images. The curation process involved removing irrelevant images and balancing the dataset across various concepts to ensure diversity. Seed images were sourced from multiple public datasets, including `ImageNet22k` and the training split of `ImageNet1k`, and expanded by retrieving similar images from web data. This meticulous curation was essential for training DINOv2 models to produce robust visual features applicable across a wide range of computer vision tasks. (Oquab et al., 2024) (Encord, 2024)

Variants of DINOv2 In fine-grained benchmarks, DINOv2 demonstrated superior performance across various tasks. The DINOv2 `ViT-g/14` model outperformed its smaller counterparts—`ViT-s/14`, `ViT-b/14`, and `ViT-l/14`—as well as other transformer-based models like `OpenCLIP`. Specifically, the `ViT-g/14` model achieved the highest average accuracy across all fine-grained benchmarks, with a classification accuracy of 92.1%. This enhanced performance underscores DINOv2's potential in applications requiring precise image understanding, such as detailed object segmentation and classification tasks, which are essential for distinguishing between different insect species. Therefore, the DINOv2 `ViT-g/14` (DINOv2-Giant) model was used in the experiment. (Oquab et al., 2024)

Comparison to ResNet-50 ResNet-50 and DINOv2-Giant differ fundamentally in their architectural design and performance. While ResNet-50 is based on convolutional layers that extract localized features through hierarchical representations, DINOv2-Giant employs self-attention mechanisms in a transformer architecture, allowing it to model long-range dependencies and capture global image relationships more effectively. This fundamental difference enables DINOv2-Giant to achieve superior performance in fine-grained classification tasks.

Despite the performance gap, ResNet-50 remains relevant due to its computational efficiency and widespread use in real-world applications. While deeper ResNet variants or alternative CNN architectures may achieve higher accuracy, ResNet-50 is supposed to serve as a representative baseline for assessing the impact of different dataset configurations on performance in fine-grained image classification.

Using Foundatoin Models Two common approaches to leverage foundation models are linear probing and fine-tuning. Linear probing involves training a classifier on fixed feature embeddings extracted from the model, offering a computationally efficient approach while still achieving strong performance. This technique was used in the conducted experiments. Fine-tuning, on the other hand, updates the model's weights on task-specific data, which is computationally more demanding but often results in higher accuracy. The choice between these methods depends on the trade-off between computational resources and performance gains. Since this work prioritizes data efficiency and explores multiple dataset variants with different foundation model and classifier combinations, features were extracted to train classifiers, following the linear probing approach.

3.1.3 Feature Reduction Techniques

Feature reduction techniques are essential in high-dimensional datasets to improve computational efficiency, reduce noise, and enhance model interpretability. Two widely used methods for dimensionality reduction are Principal Component Analysis (PCA) and Uniform Manifold Approximation and Projection (UMAP).

Principal Component Analysis (PCA) Principal Component Analysis (PCA) is a linear dimensionality reduction technique that projects the original feature space onto a lower-dimensional subspace while maximizing variance. It is commonly used for feature reduction, helping to remove noise while retaining most of the relevant information in the data. PCA relies on linear transformations, meaning it represents data as a weighted sum of original features without introducing non-linear mappings. While effective for many datasets, this limitation makes PCA less suitable for capturing complex, non-linear relationships within the data. (Kurita, 2021)

Uniform Manifold Approximation and Projection (UMAP) Uniform Manifold Approximation and Projection (UMAP) is a non-linear dimensionality reduction technique designed to preserve both local and global structures in data. Unlike PCA, which relies on linear transformations, UMAP constructs a high-dimensional graph representation of the data and optimizes a low-dimensional embedding that maintains the original structure.

UMAP is particularly effective for visualization and clustering tasks, as it captures complex, non-linear relationships that PCA cannot. However, as it is a newer technique, it requires careful selection of hyperparameters such as the number of neighbors and minimum distance. The choice of parameters can make UMAP more challenging to apply effectively compared to PCA, which has a more straightforward implementation and interpretation. (Ghojogh et al., 2021)

3.2 Hold-Out Method and Cross-validation

A fundamental step in model evaluation is assessing generalization performance, which ensures that a trained model performs well on unseen data. The hold-out method is the most straightforward approach, where the dataset is split into separate training, validation, and test sets (e.g., 80%-10%-10%). The model is trained on the training set, hyperparameters are tuned on the validation set, and final performance is assessed on the test set. This method is computationally efficient and was also used for preliminary testing and final experiments. However, a key drawback of the hold-out method is its dependency on a single data split, which may not fully capture dataset variability, leading to high variance in performance estimates and a risk of overfitting to the specific partition.

To obtain a more robust evaluation, cross-validation is often preferred, particularly when dataset size is limited. K-fold cross-validation is a widely used technique where the dataset is divided into K equally sized folds. The model is trained on $K-1$ folds and validated on the remaining fold, repeating the process K times so that each fold serves as a validation set once. The final performance metric is obtained by averaging across all folds. Standard choices are 5-fold or 10-fold cross-validation, which balance computational cost and result stability. More folds (e.g. 10 or more) are particularly useful for smaller datasets, as they maximize the training data per iteration, while fewer folds (e.g. 5) are often sufficient for larger datasets to reduce computational overhead. (James et al., 2013)

Stratified K-Fold Cross-Validation In standard K-fold cross-validation, folds are created through random splits, which can lead to an uneven class distribution, particularly in imbalanced datasets. This can result in misleading performance estimates if certain classes are underrepresented in some folds. Stratified K-fold cross-validation addresses this issue by ensuring that each fold maintains approximately the same class distribution as the full dataset. By preserving class ratios, stratification prevents models from becoming biased toward more frequent classes, leading to more reliable and generalizable results. (Berrar et al., 2019) While the

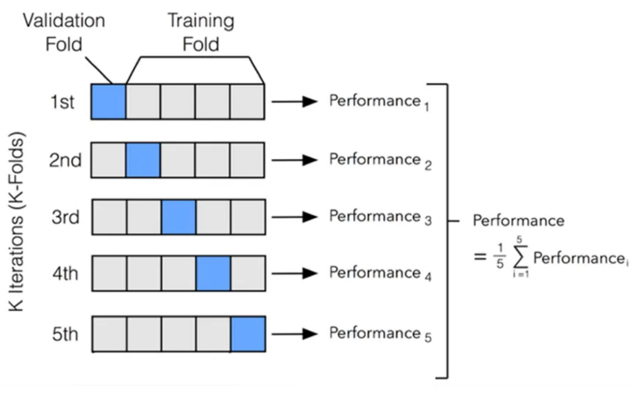


Figure 1: Visualization of 5-fold cross-validation. (Shen, 2020)

dataset used in this experiment was not imbalanced, stratification was still applied to ensure an equal distribution and mitigate a potential source of error.

Overall, while the hold-out method provides a quick estimate of performance, cross-validation — especially stratified K-fold—is the preferred approach for ensuring robust evaluation, particularly when dealing with fine-grained datasets where an equal distribution of samples per class is desired.

3.3 Classifiers

Foundation models provide the basis for extracting valuable features from data, which can then be used as inputs for classifiers like **K-Nearest Neighbors (KNN)** or **Logistic Regression (LR)** to perform classification tasks. These classifiers differ in how they define and use decision boundaries to separate data into distinct classes.

3.3.1 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a popular and widely used algorithm in the field of supervised learning. It is known for its simplicity, interpretability, and effectiveness in various machine learning tasks. KNN is a non-parametric and lazy learning algorithm, meaning it does not make any underlying assumptions about the data distribution and does not perform any explicit training phase. Instead, it simply stores the entire training data set and makes predictions at runtime based on the similarity between data points.
KNN can be used for both regression and classification tasks, although it is more commonly applied to classification problems. The core idea of KNN is to classify a new data point based on the labels of its k nearest neighbors in the feature space. (Müller and Guido, 2016)

- **Distance Metrics:** To determine the nearest neighbors, KNN relies on distance metrics such as Euclidean distance, Manhattan distance, or Minkowski distance. The choice of distance metric can affect the performance of the algorithm. The Euclidean distance between two points x_i and x_j in a feature space can be calculated as follows:

$$d(x_i, x_j) = \sqrt{\sum_{n=1}^{N} (x_{i,n} - x_{j,n})^2}$$

  Where N is the number of features.

- **Plurality Voting:** Once the k nearest neighbors are identified, the algorithm uses plurality voting to decide the class label of the new data point. Each neighbor "votes" for its class, and the class with the most votes is assigned to the new data point. The number of neighbors participating in the voting can be specified in the algorithm's parameters and can also strongly influence its performance. (IBM, 2025)

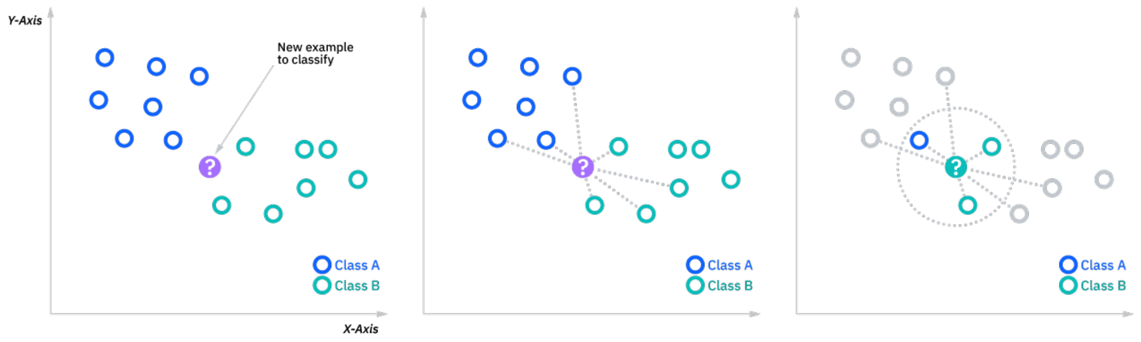


Figure 2: Diagram showing new sample placed into feature space and following classification as class B with $neighbors = 3$ observable on right hand side. (IBM, 2025)

KNN is an interesting choice for comparison with Logistic Regression due to their fundamentally different working mechanisms. While KNN operates on the principle of feature similarity, assigning labels based on the closest data points, Logistic Regression fits a hyperplane to separate probability distributions. In the following its working mechanisms will be elaborated in greater detail.

3.3.2 Logistic Regression

Logistic Regression (LR) is a widely-used example of a linear classifier, commonly applied to classification tasks. Despite its name, it is not a regression algorithm but a classification method that predicts the probability of a data point belonging to a particular class. It works by learning a linear decision boundary in the feature space, which separates one class from another (in binary classification) or distinguishes among multiple classes (in multi-class classification). The outputs are transformed to probabilities through the application of the **softmax function**.

- **Linear Transformation:** LR begins by applying a linear transformation to the input features. For each class i, the algorithm computes a score (called a *logit*) as:

  $$z_i = \mathbf{w}_i \cdot \mathbf{x} + b_i$$

  where $\mathbf{x}$ is the input feature vector, $\mathbf{w}_i$ are the weights for class i, and b_i is the bias term. (Hartmann et al., 2023)

- **Softmax Activation:** In the case of multi-class classification, the logits z_i are converted into probabilities using the softmax function:

  $$P(\text{class}_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

  where C is the total number of classes. The probabilities indicate how likely the input belongs to each class. (Stanford, 2025)

- **Loss Function:** LR uses **cross-entropy loss** to measure performance by comparing predicted probabilities to actual class labels:

  $$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij} \log P_{ij}$$

  where:

  - N is the number of samples,
  - C is the number of classes,
  - y_{ij} is 1 if sample i belongs to class j (and 0 otherwise),
  - P_{ij} is the predicted probability that sample i belongs to class j.

  High confidence in correct predictions results in lower loss, as $\log(P_{ij})$ approaches 0 when P_{ij} is close to 1 for the true class. Conversely, incorrect or uncertain predictions increase the loss. The total loss is averaged across all samples, and training minimizes this loss by adjusting the model's parameters. (Hartmann et al., 2023)

- **Optimization:** To learn the weights $\mathbf{w}_i$ and biases $\mathbf{b_i}$, the algorithm minimizes the cross-entropy loss by iteratively adjusting the parameters using optimization techniques such as gradient descent, which updates parameters in the direction of the steepest decrease in the loss function. More advanced optimizers like **Adam** can also be used, as they combine the benefits of **momentum**, which smooths updates by considering past gradients to reduce oscillations, and **adaptive learning rates**, which automatically adjust the step size for each parameter based on the magnitude of recent gradients, often resulting in faster and more efficient convergence. Through this process, LR finds a decision boundary that best separates the classes in the feature space. (Kingma and Ba, 2014)

Learning Rate Scheduling Learning rate scheduling can be an important technique to improve the performance of a model. It dynamically adjusts the learning rate during training to improve convergence and model performance.

The learning rate is a crucial hyperparameter that determines the step size at each iteration while moving toward a minimum of the loss function. If the learning rate is too high, the model may overshoot the minimum, while if it's too low, training may be slow or get stuck in local minima.

Learning rate scheduling addresses these issues by adapting the learning rate throughout the training process. Common strategies include:

- **Step Decay:** Reduce the learning rate by a factor after a fixed number of epochs. (PyTorch, 2025c)
- **Exponential Decay:** Continuously decrease the learning rate exponentially over time. (PyTorch, 2025a)
- **Reduce on Plateau:** Lowering the learning rate when the model's performance stops improving for a predefined number of epochs. It is particularly useful when training stagnates, helping the optimizer escape shallow local minima and continue converging towards a better solution. (PyTorch, 2025b) This method was tried in the preliminary tests described in Section 5.2.

The discussed techniques were applied to the dataset which will be introduced and described in the following.

4 Dataset

As highlighted earlier, citizen science projects can play a crucial role in advancing biodiversity research. The Global Biodiversity Information Facility (GBIF) is an international initiative that provides open access to biodiversity data from around the globe. It aggregates information from a diverse array of sources — including citizen science, research institutions, and natural history collections — to support conservation efforts and ecological monitoring. (gbi, 2025) A prime example of this collaborative approach is Observation.org, a citizen science platform, where users contribute observations enriched with images and metadata (obs, 2025). These submissions undergo expert verification before becoming publicly accessible and are subsequently integrated into larger datasets, such as those hosted by GBIF.

4.1 Dataset Origin

The dataset used in this experiment was created using GBIF and is accessible at https://doi.org/10.15468/dl.8byj47. It integrates data from Observation.org, which collects and verifies biodiversity observations submitted by contributors worldwide.

To ensure relevant data selection for the purpose of fine-grained image classification, filtering was applied based on the following criteria:

- **Lifestage**: Imago (Fully developed insects)
- **Media Type**: StillImage (Images)
- **Scientific Name**: *Lepidoptera* (Butterflies and moths)

Taxonomic Hierarchy *Lepidoptera*, an order within the class *Insecta*, includes both butterflies and moths. This diverse group is organized hierarchically into several taxonomic groups, such as superfamilies, families, genera, and species. For example, the superfamily *Papilionoidea* encompasses all true butterflies, which includes families such as *Pieridae* (whites and sulphurs). Moving down the hierarchy, the genus *Pieris* is part of the family *Pieridae*, and within this genus, species like *Pieris napi* (green-veined white butterfly) or *Pieris rapae* (small white butterfly) are identified. (Heppner, 2010) In the experiment, the focus lies on the classification at the species level.

4.2 Dataset Structure

The dataset downloaded from GBIF contains structured information about each observation, recorded in files such as `verbatim.txt`, which includes 190 columns covering a wide range of aspects, from taxonomy groups and geographic data to metadata related to data usage and legal considerations. Instead of directly providing the images, links to them are listed in the `multimedia.txt` file under the column `identifier`. Each sample is identified by a unique `gbifID`, which serves as a key linking corresponding samples across different files.

Upon downloading, the dataset (which doesn't directly include image data) consists of 3.67 million samples covering 7,664 distinct species, with a strong class imbalance, as shown in Figure 3. Specifically, the 589 species with more than 1,000 samples account for approximately 87.84% of the entire dataset. To mitigate class imbalance and establish a more reliable base by ensuring a minimum number of samples per species, the dataset was filtered to retain only these 589 species as illustrated. Each species in the final dataset contains between 1,000 and 3,000 samples. This selection allows for the construction of **nine balanced dataset variants** as shown in Table 1, which are the foundation to examine the effects of different sample and species counts. This approach smallers the number of image downloads while attempting to make the most possible use of the initial dataset.

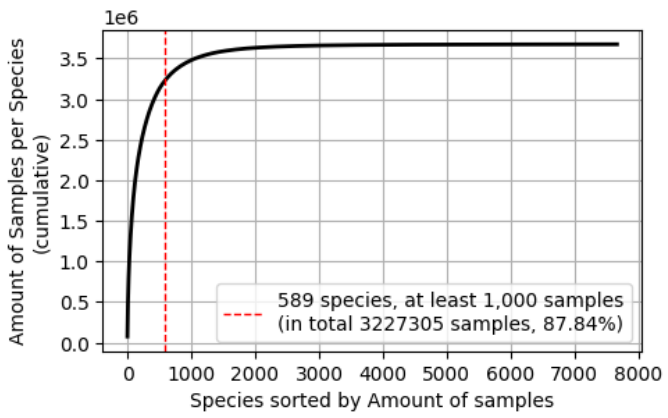


Figure 3: Cumulative plot showing imbalance in the amount of samples per species.

Sample Amount per Species: **Species Amount**	**500**	**1,000**	**2,000**	**3,000**
277	138,500	**277,000**	554,000	**831,000**
387	193,500	387,000	774,000	-
589	294,500	**589,000**	-	-

Table 1: Total amounts of samples for the nine different dataset variants used in the experiment. The dataset configurations marked in bold are investigated in greater detail than the other variations in Section 6.

Image Downloading Process Another factor benefiting from the reduction in sample size is the time required to download images from the links provided in `multimedia.txt`. Since no batch-download functionality is available to date, the 1.39 million images had to be retrieved sequentially from Observation.org. In retrospect, the amount of images which has been downloaded was more than really necessary, since for species containing between e.g. 2,000 and 3,000 samples, all available images for that species were downloaded as shown in Figure 4.

To prevent server overload, a download speed limit of two images per second was established in coordination with Observation.org. This process spanned approximately 10 days, including minor interruptions such as network downtime or broken links. The complete dataset comprises a total of 265 GB of images, resulting in an average file size of approximately 200 kB per image.

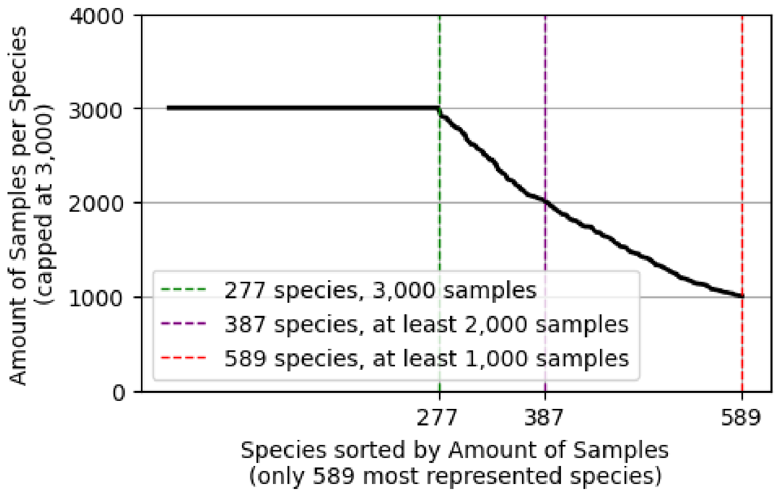
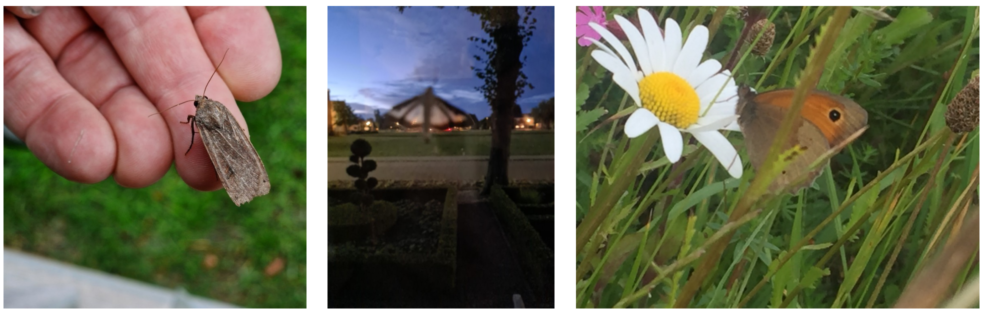


Figure 4: Figure showing the sample amounts per species that were downloaded.

4.3 Challenges in Citizen Science Image Processing

Since citizen science datasets rely on contributions from numerous volunteers, the quality of individual samples varies significantly, causing potential for error in several ways:

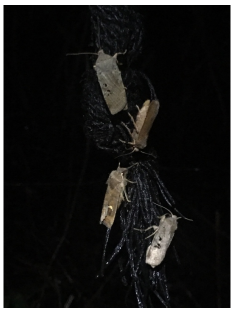
- **Variability in image quality**: Differences in camera resolution, image format (resulting in later cropping), and lighting conditions, such as brightness, sharpness, and contrast, lead to inconsistencies. Additionally, variations in size and angle in which specimens are captured or obstacles impairing clear vision on the object can make identification difficult. Examples for those cases are shown in Figure 5.



(a) (sam, s) (b) (sam, p) (c) (sam, q)

Figure 5: Samples with varying image quality, brightness, focus as well as impaired visibility and size of the insect in the image.

- **Multiple specimens**: Some images contain multiple insects, occasionally even from different species. An example is shown in Figure 6 `image a`. This introduces ambiguity and increases the risk of misclassification, as the model may not correctly associate the provided label with the intended specimen. This issue is particularly prevalent in moth images, as some are taken from moth traps, which often contain numerous individuals and species as shown in Figure 6 `image b`. Traps however are presumably rather rarely found in the dataset since multiple random sample slices consisting of 1,000 images only revealed one trap image.


(a) (sam, r)

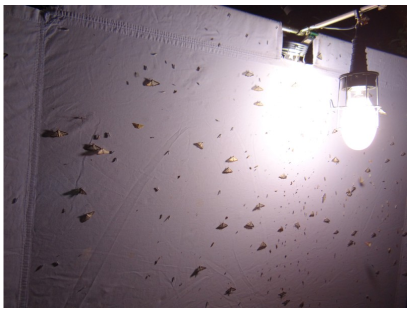

(b) (sam, t)

Figure 6: Samples containing multiple insects.

- **Common genera**: Species within the same genus, such as those in the genus *Eupithecia* (see Figure 7), often exhibit only subtle morphological differences, which makes distinguishing between them a challenging task. The dataset includes 589 species, which are classified into 401 genera. Among these, 301 genera are represented by a single species, while the other 100 genera contain more than one species. Those are later on referred to as (genus) clusters. In total, 288 species belong to these 100 clusters. The size of each genus cluster is defined by the number of species it contains, and larger clusters present a higher potential for confusion. Table 2 provides the distribution of all $301 + 288 = 589$ species, while the mentioned 288 species belong to a genus cluster that contains at least two species.

Cluster Size (CS)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**Sum**
Amount of Genera	301	57	23	9	6	3	0	0	0	1	1	401
Amount of Species	301	114	69	36	30	18	0	0	0	10	11	589

Table 2: The table shows the the distribution of 589 species across 401 different genera. For example the column for CS of 3 means, that there exist 23 genus clusters that each contain 3 species. In total $23 \times 3 = 69$ species belong to a genus cluster with a CS of 3.

- **Genus labels for unclassified species**: Additionally, some samples could not be assigned to a specific species by experts with certainty and were therefore labeled after the genus (e.g. *Eupithecia Spec.*). These *Spec.*-labels, which encompass a variety of unclassified species belonging to the same genus, introduce a significant risk of misclassification. As a classifier attempts to identify the correct species instead of assigning the sample to the *Spec.*-label, it is unable to correctly predict the species even though the classification might technically be correct. There exist 10 *Spec.*-labels, of which 5 are the only species for their corresponding genus. The other 5 *Spec.*-labels belong to 5 genus clusters of different sizes, accounting for 30 species in total (including the *Spec.*-labels). These labels are also included in the 288 species, belonging to a genus cluster as introduced before. Ideally, those *Spec.*-labels should have been removed before classification, but this issue was only discovered retrospectively. As a result, their impact on model performance will be analyzed in the in Section 6.4.3.

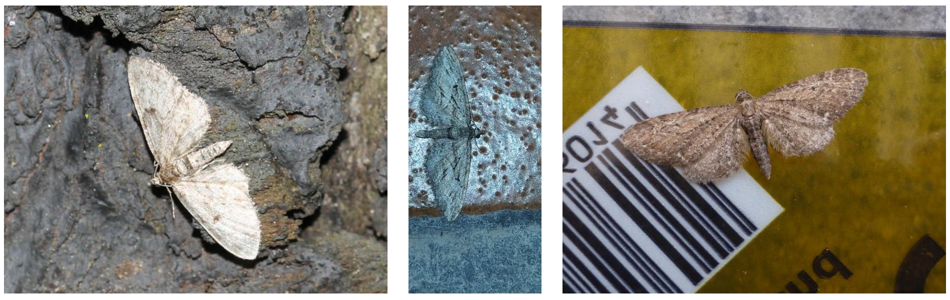

(a) (sam, m) (b) (sam, n) (c) (sam, o)

Figure 7: Examples of visually similar species belonging to the same genus: (a) *Eupithecia Spec.* which contains unclassified species belonging to the *Eupithecia* genus, (b) *Eupithecia oxycedrata* and (c) *Eupithecia vulgata*

4.4 Quality Control Strategies

Several approaches were tested to improve dataset quality, though most did not yield satisfactory results. The following strategies were explored:

1. **Filtering Dark or Underexposed Images**: Pixel brightness was analyzed to exclude excessively dark images. Images below or above a threshold were either blacklisted or flagged for manual review. However, many moth images were incorrectly flagged because their backgrounds were entirely black while e.g. the moth itself remained clearly visible. As a result, this method only identified 16 genuinely unusable images which were filtered out, making the filtering process insignificant for dataset improvement.

2. **Detecting the Number of Insects Per Image**: OpenCV-based techniques were tested to filter out images containing multiple or overly small insects. However, due to the high detail in many images, contour-based object detection was found ineffective.

3. **Checking for Synonymous Species**: Due to variations in discoverers and naming conventions, species often have multiple synonyms, leading to cases where different names refer to the same species. To prevent misclassification, the dataset was checked for synonyms before data retrieval to ensure that no duplicate species were included. However, this process did not reveal the 10 *Spec.*-labels that represent genera rather than distinct species as already explained in 4.3.

In conclusion, given the impracticality of manually reviewing and curating 1.39 million images within a reasonable time frame, the dataset curation attempt is considered unsuccessful. However, this limitation presents an opportunity to assess the generalizability of the foundation models used for feature extraction. The dataset, despite its imperfections, serves as a real-world test for data efficiency and may provide valuable insights for future classification systems developed for or based on citizen science projects.

5 Experimental Setup

Concerning hardware, the experiment was conducted using the workstations at Otto Friedrich University. For the tests, the following hardware was utilized:

- **Operating System & Architecture:** Ubuntu 20.04.1, x86_64
- **CPU:** Intel Xeon W-2265, 12 cores, 24 threads, 3.50 GHz (Max: 4.8 GHz)
- **GPU:** 2× NVIDIA RTX A5000, 24 GB VRAM
- **Memory (RAM):** 128 GB

The most important libraries and methods shown in Table 3 were implemented using Python version 3.8.10.

For further insight, the source code is available on GitHub:
https://github.com/leo-grz/BachelorThesis_LepidopteraFGIC.git.

Library	**Version**	**Used Components**
scikit-learn	1.3.2	`train_test_split`, `PCA`, `StratifiedKFold`, `KNeighborsClassifier`, `accuracy_score`, `precision_score`, `recall_score`, `f1_score`, `predict_proba`
PyTorch	2.4.1	`DataLoader`, `CrossEntropyLoss`, `Adam`, `ReduceLROnPlateau`, `Linear`, `softmax`, **DINOv2-Giant (vitg14)**
Torchvision	0.19.1	**ResNet-50 (ImageNet1k_V2)**

Table 3: Libraries and components used for the implementation

5.1 Feature Extraction

After selecting the species and retrieving the corresponding image samples, further preprocessing steps were necessary before training the classifiers. Since the objective was to leverage the foundation models ResNet-50 and DINOv2-Giant (in the following referred to as DINOv2) the next step involved extracting the feature embeddings.

All images were resized to 224×224 pixels (3 color channels) before being converted into tensors. For feature extraction, ResNet-50 was modified by replacing its fully connected classification layer with an identity mapping to obtain the 2048-dimensional feature embeddings. Similarly, DINOv2 was adapted by removing its classification head, resulting in 1536-dimensional feature embeddings per image.

The extracted embeddings were stored along with the corresponding globally unique `gbifID` (for later traceability) and numerical class labels (ranging from 0 to 588). This resulted in two large datasets:

- ResNet-50 embeddings: 1.39 million samples, each represented by a 2048-dimensional feature vector, stored in a 7.4 GB lossless compressed `.npz` file.
- DINOv2-Giant embeddings: 1.39 million samples, each represented by a 1536-dimensional feature vector, stored in a 5.5 GB lossless compressed `.npz` file.

After verifying the uniqueness of each `gbifID` and the correct label encoding, both datasets were used to create nine dataset variants each as described in Table 1 resulting in 18 different dataset variations.

5.2 Preliminary Tests

Before conducting cross-validation to determine the best-performing classifiers, preliminary experiments were necessary to assess potential feature reduction techniques and establish reasonable baseline hyperparameters.

Feature reduction was performed using two different techniques: Uniform Manifold Approximation and Projection (UMAP) and Principal Component Analysis (PCA) as introduced in Section 3.1.3. In the case of UMAP, the reduced feature representations were evaluated by varying both the number of neighbors and the target dimensionality. For PCA, the reduced representations, known as principal components, were assessed by modifying the number of retained components.

The impact of these dimensionality reductions was analyzed in the context of the two classifiers K-Nearest Neighbors (KNN) and Logistic Regression (LR), as introduced in Section 3.3

Choice of Parameter Configuration To assess the impact of different feature reduction techniques and establish suitable hyperparameter ranges, a series of tests were conducted using a wide span of hyperparameter configurations for all feature reduction methods and classifiers. Hereby each classifier with all different combinations of hyperparameters was trained on the UMAP embeddings or principal components (reduced using PCA).

For KNN, 10 different values for k were tested. For LR, a total of 20 combinations of learning rate and epochs were considered. This resulted in $10 + 20 = 30$ classifier configurations tested for each feature reduction setup.

With 6 different UMAP configurations and 3 PCA configurations, a total of $(6+3) \times 30 = 270$ runs were conducted for each dataset variant. Since both ResNet-50-based and DINOv2-based datasets were evaluated separately, this led to a combined total of $270 \times 2 = 540$ experimental runs for preliminary testing. The tested hyperparameter ranges are summarized in Table 4.

Choice of Dataset for preliminary Tests All preliminary tests were conducted using the two dataset variants (ResNet-50-based and DINOv2-based) consisting of 277 species with 1,000 samples per species resulting in a total amount of 277,000 samples each. This configuration was selected for two reasons:

- Using a dataset with a limited total sample size (277,000) compared to other configurations allows for a faster hyperparameter search, enabling the evaluation of a broader range of feature reduction settings and classifier configurations within a reasonable time frame.

- The selected dataset serves as a balanced benchmark: it enables a classification task that is neither trivially easy (due to an excessive number of samples) nor excessively difficult (due to the lowest amount of species within the different dataset configurations). Additionally, it allows for easier comparisons across different species counts, as 1000 samples is the maximum number for which all species have sufficient data.

The dataset was split using the *hold-out* method (as introduced Section 3.2 with an 80/10/10 train/validation/test ratio for LR and 80/20 train/test ratio for K-Nearest Neighbors. Stratification was applied to ensure an equal distribution of samples across each split.

Parameter	**Values**
Dataset Variants	ResNet-50-based and DINOv2-based 277 Species with 1,000 Samples each
PCA: Reduced Feature Size	{256, 512, 1024}
UMAP: Reduced Feature Size	{128, 512, 1024}
UMAP: Number of Neighbors	{50, 100}
KNN: Number of Neighbors	{1, 3, 5, 10, 50, 100, 300, 500, 700, 1000}
LR: Learning Rate	{0.1, 0.01, 0.001, 0.0001}
LR: Number of Epochs	{250, 500, 750, 1000, 1500}

Table 4: Hyperparameter ranges used for the initial evaluation of UMAP and PCA feature reduction, as well as baseline parameter discovery.

Results of Preliminary Tests Based on the results of the preliminary experiments, the optimal feature reduction method, reduction extent, and classifier hyperparameters were selected for the main cross-validation tests.

Figure 8 visualizes the classification accuracy achived by KNN and LR on feature embeddings (extracted from ResNet-50 or DINOv2) reduced by using PCA and UMAP. Each box represents the accuracy distribution across KNN and LR derived with varying combinations of hyperparameters.

The results clearly indicate that PCA outperforms UMAP across both dataset variants, with PCA-based reductions yielding consistently higher classification accuracies. As visualized, the range of tested hyperparameters and classifiers is causing substantial variations in accuracy but the significantly higher means highlight a noticeable difference between UMAP and PCA: approximately 58% difference in accuracy for the DINOv2-based dataset and 44% difference for the ResNet-50-based dataset (compare Figure 8).

Given those results, PCA is the preferred feature reduction method for further cross-validation experiments and all subsequent results discussed further on.

Among the three tested numbers of principal components (256, 512, and 1024), a slight performance improvement was observed with an increasing number of features.

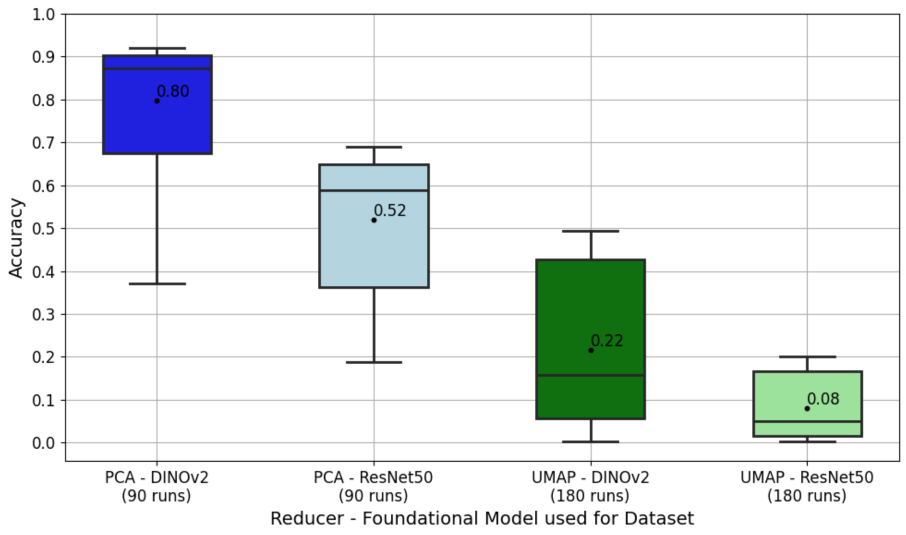


Figure 8: Box plot showing the distribution of accuracies achieved by KNN and LR for different combinations of reducers and foundation models used in dataset creation. The horizontal line in each box represents the median accuracy, each point represents the mean accuracy.

The accuracy with 256 principal components is noticeably lower compared to the results obtained with 512 and 1024 components. This is illustrated in Figure 9 and Figure 11, where the blue and green lines (representing 512 and 1024 components) are noticeably closer to each other than the red line (256 components) is to either of them.

Despite the slight increase in accuracy with 1024 components, 512 was chosen due to its significantly lower training time, averaging approximately 29 seconds compared to 53 seconds for 1024 components. These times were computed across all KNN and LR runs (with 1500 epochs) for both (DINOv2-based and ResNet-50-based) datasets per amount of principal components. It is important to note that training duration increases substantially with larger datasets, as the reported averages are based on the third smallest dataset (containing 277,000 samples).

Since the objective of this test is to identify the most optimal hyperparameter combination for cross-validation (which multiplies the number of runs by a factor of k in k-fold CV), lower computation time was preferred over a marginally higher accuracy.

For the cross-validation tests, a learning rate of 0.001 with 1500 epochs was chosen as the starting point. A lower learning rate helps ensure that the model converges more steadily and does not overlook important patterns in the data, reducing the risk of overshooting the optimal solution. However, lower learning rates require more training epochs to reach convergence, which is why 1500 epochs were selected.

As observable in Figure 10, validation loss and accuracy converge differently for the two dataset types. For both learning rates 0.1 and 0.01 (represented by the blue and yellow lines), the model overshoots, resulting in an increase in loss after an initial drop. However, for learning rates 0.001 and 0.0001 (represented by the green and

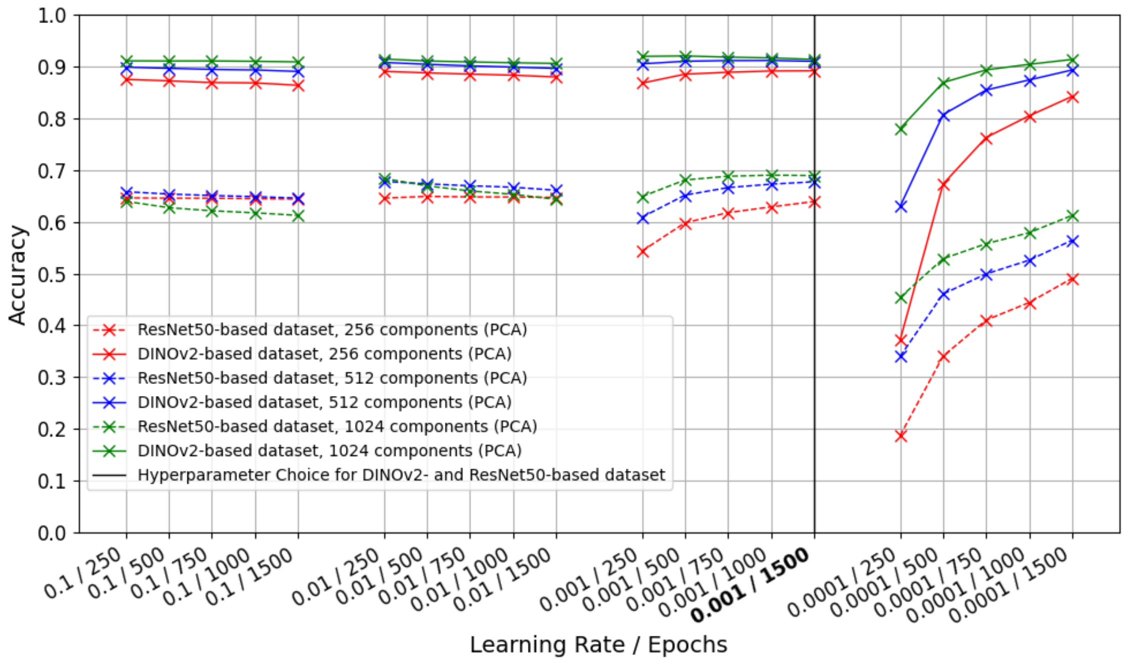


Figure 9: Lineplots showing the accuracies achieved by the Logistic Regression Classifier with 20 different combinations of learning rate and epochs for two different datasets types whos' embeddings were reduced to different amounts using PCA.

red lines), the loss consistently decreases while accuracy increases, with convergence reached significantly earlier on the DINOv2-based dataset. At a learning rate of 0.001, the loss reaches its lowest point for both datasets, while a lower learning rate (0.0001) might achieve a similar minimum with additional epochs. This trade-off was further explored by testing surrounding learning rates (0.005, **0.001**, 0.0005) and increasing epochs (**1500**, 2000) in the cross-validation experiment as listed in Table 5.

Learning Rate Scheduling It is worth noting that Learning Rate Scheduling was also explored in a sampling manner. Starting with an initial learning rate of 0.001, a *Reduce On Plateau* scheduling strategy was employed (as discussed in Section 3.3.2). The learning rate was gradually decreased to 10^{-6}, with multiple factor values (0.95, 0.9, 0.5, 0.1) for learning rate multiplication. Various thresholds (0.1, 0.01) were tested to determine the required difference for adjusting the learning rate, along with different patience values (3, 5, 10, 20) to specify the number of epochs the threshold must not be exceeded. Despite these efforts, the accuracy did not surpass the results obtained using a fixed learning rate of 0.001, and for simplicity, further exploration of learning rate scheduling was discontinued.

For KNN, using 50 neighbors resulted in the highest accuracy for both datasets. However, accuracy trends differed between the two: while the DINOv2-based dataset showed stable accuracy between 10 and 50 neighbors, the ResNet-50-based dataset exhibited a slight improvement at 50 neighbors before declining.

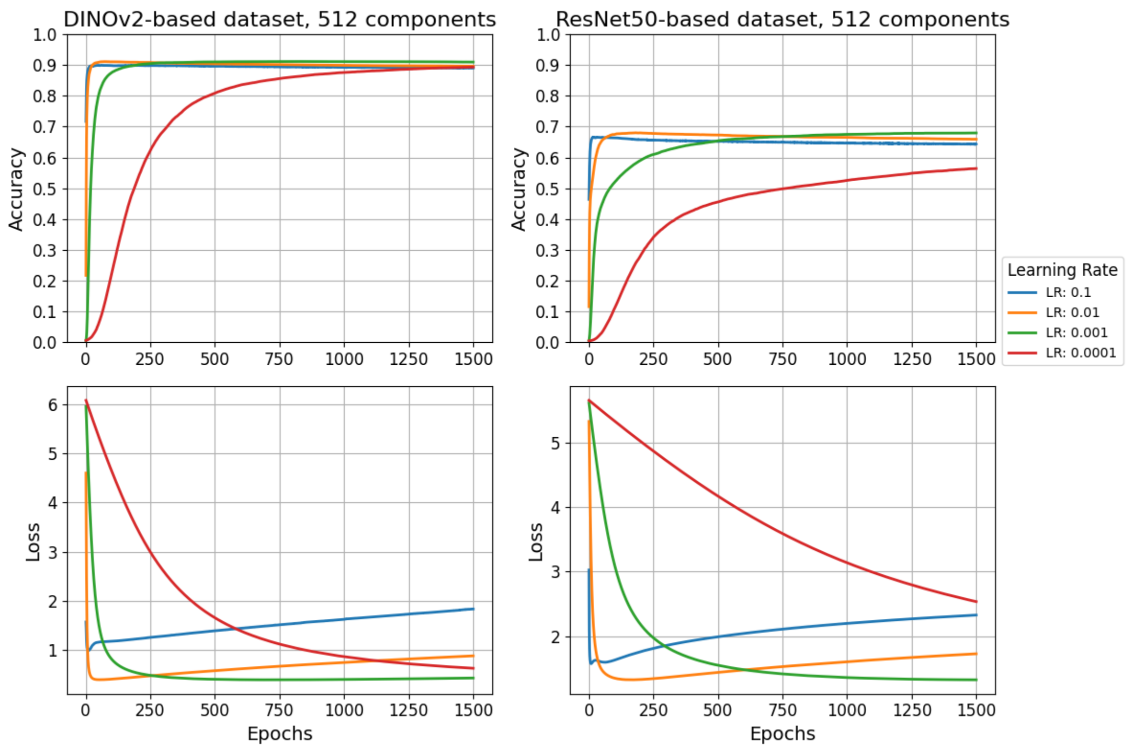


Figure 10: Loss and accuracy for different learning rates with 1500 epochs on embeddings reduced to 512 components using PCA

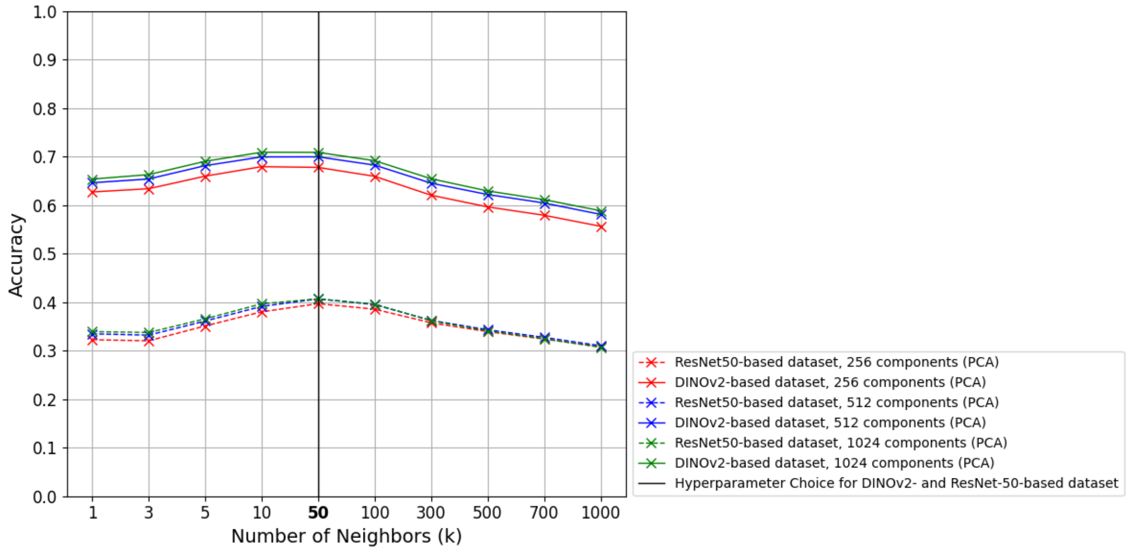


Figure 11: Accuracies achieved by the KNN Classifier with 10 different values for k on two different datasets types. The embeddings were reduced using PCA.

Due to these differences, the hyperparameter search ranges in cross-validation were adjusted accordingly:

- For the DINOv2-based dataset, the chosen values were $k = 20, 35, 50$.
- For the ResNet-50-based dataset, since 50 neighbors provided the peak accuracy, a slightly shifted range of $k = 35, 50, 65$ was used.

This selection ensured that each model was evaluated in its most relevant range.

5.3 Cross-Validation Tests

Based on the results of the preliminary tests, the following set of parameters and methods was evaluated using stratified 5-fold cross-validation. The goal was to refine the selection of optimal hyperparameters by testing a smaller, more targeted range derived from previous experiments, as cross-validation results are more reliable and robust, providing credibility for small changes.

Unlike the preliminary tests, this cross-validation experiment was conducted across all 18 dataset configurations (9 ResNet50-based and 9 DINOv2-based), employing stratified 5-fold cross-validation for every hyperparameter combination listed in Table 5.

With three different hyperparameter configurations for KNN and six for logistic regression, each averaged over the 5 folds for each of the 18 dataset configurations, a total of

$$(3_{\text{KNN}} + 6_{\text{LogReg}}) \times 5_{\text{splits}} \times 9_{\text{datasets}} \times 2_{\text{foundation models}} = 810 \tag{1}$$

runs were performed.

Parameter	**Values**
PCA: Reduced Feature Size	512
KNN: Number of Neighbors (ResNet-50)	{35, **50**, 65}
KNN: Number of Neighbors (DINOv2)	{20, 35, **50**}
Logistic Regression: Number of Epochs	{**1500**, 2,000}
Logistic Regression: Learning Rate	{0.005, **0.001**, 0.0005}

Table 5: Hyperparameter ranges used for the cross-validation experiment. Marked bold are the best hyperparameters derived from preliminary experiments.

Choice of Hyperparameters for Further Analysis The optimal hyperparameters were selected based on mean accuracy and standard deviation across all nine dataset configurations for each foundation model. The results were averaged over five cross-validation folds to ensure robustness.

Dataset Type	Learning Rate (LR)	Epochs (LR)	Neighbors (KNN)
ResNet-50-based	0.001	1500	50
DINOv2-based	0.0005	1500	35

Table 6: Selected hyperparameters for further analysis: Learning Rate and Epochs (Logistic Regression) and Neighbors (KNN).

The choices were guided by a trade-off between accuracy and stability across the dataset variants, balancing mean accuracy with standard deviation (std). All tables and figures discussed in the following can be found in the appendix.

- **Logistic Regression on DINOv2-based datasets:** Learning rate of 0.0005 with 1500 epochs achieved the highest mean accuracy (90.00%) with the lowest standard deviation (1.99%). Increasing to 2,000 epochs slightly reduced stability without accuracy gain as shown in Figure 29. (Table 12)
- **KNN on DINOv2-based datasets:** $k = 35$ was selected as it provided a balance between mean accuracy (68.43%) and stability (4.41% std). Although $k = 20$ had slightly higher mean accuracy (68.59%), it also had a higher standard deviation (4.47%) (Table 13).
- **Logistic Regression on ResNet-50-based datasets:** Learning rate of 0.001 with 1500 epochs were chosen for its stable accuracy (64.82%, std 4.16%). While 2,000 epochs slightly improved accuracy (64.90%), it increased the deviation (4.42%) and increased validation loss for some dataset variants, as demonstrated in Figure 31. (Table 14)
- **KNN on ResNet-50-based datasets:** $k = 50$ was selected for its balance of accuracy (38.19%) and stability (5.03% std). The highest accuracy (38.31% at $k = 35$) had the highest standard deviation (5.14%), making $k = 50$ the optimal choice. (Table 15).

All subsequent analyses, including confidence evaluation, are derived using the hyperparameters shown in Table 6.

6 Results & Analysis

6.1 Cross-Validation Results

6.1.1 Accuracy & other Performance Metrics

As illustrated in Figure 12, classifiers trained on DINOv2 embeddings consistently achieve higher accuracy than those trained on ResNet-50 embeddings. Additionally, Logistic Regression (LR) outperforms KNN across both dataset types. On average, the accuracy gap between KNN and LR is 21.56% for the DINOv2-based dataset and 26.61% for the ResNet-50-based dataset, as the gaps stays relatively consistent across all dataset variants. Furthermore, classification accuracy improves with an increasing number of samples per species, while it decreases as the total number of species increases.

In future, the combinations between foundation models (FM) used for feature extraction and classifiers (CLF) will be referred to as **FM-CLF** combinations like **DINOv2-KNN** or **ResNet-50-LR**.

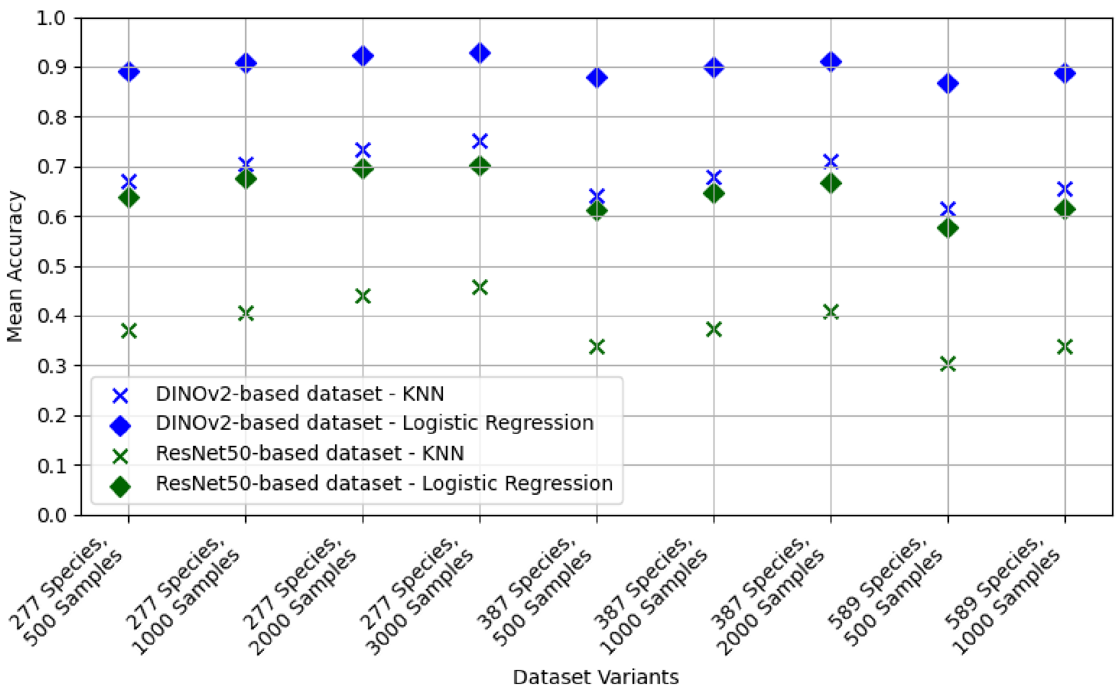


Figure 12: Top-1 accuracies derived using LR and KNN with cross-validation on the 18 different dataset variants with hyperparameters listed in Table 6. The exact mean accuracies shown in the Figure are listed in the appendix under Table 16.

Precision, Recall and F1-Score in Comparison to Accuracy To assess the model's consistency, the minimum, mean, and maximum differences between key performance metrics (precision, recall, and f1-score) and accuracy were calculated. The mean difference of less than 0.1% between these metrics and accuracy indicates highly consistent performance across different evaluation criteria. This consistency suggests that the models do not disproportionately favor either false positives or false negatives, pointing to a well-balanced classification performance. The exact numerical values can be found in the appendix under Table 17.

Standard Deviation of Metrics Across Folds To assess the impact of the dataset split on classifier performance, the standard deviation of accuracy over the five splits was computed for each of the $(1_{\text{KNN}} + 1_{\text{LogReg}}) \times 18_{\text{datasets}} = 36$ cross-validation runs. The mean standard deviation across all setups is 0.15%, with a maximum of 0.32% and a minimum of 0.04%. Similarly precision, recall and f1-score only show marginal changes in between the splits comparable to the accuracy. Minimum, mean and maximum standard deviations can be accessed in Table 18 in the appendix.

6.1.2 Computational Efficiency of Classifiers

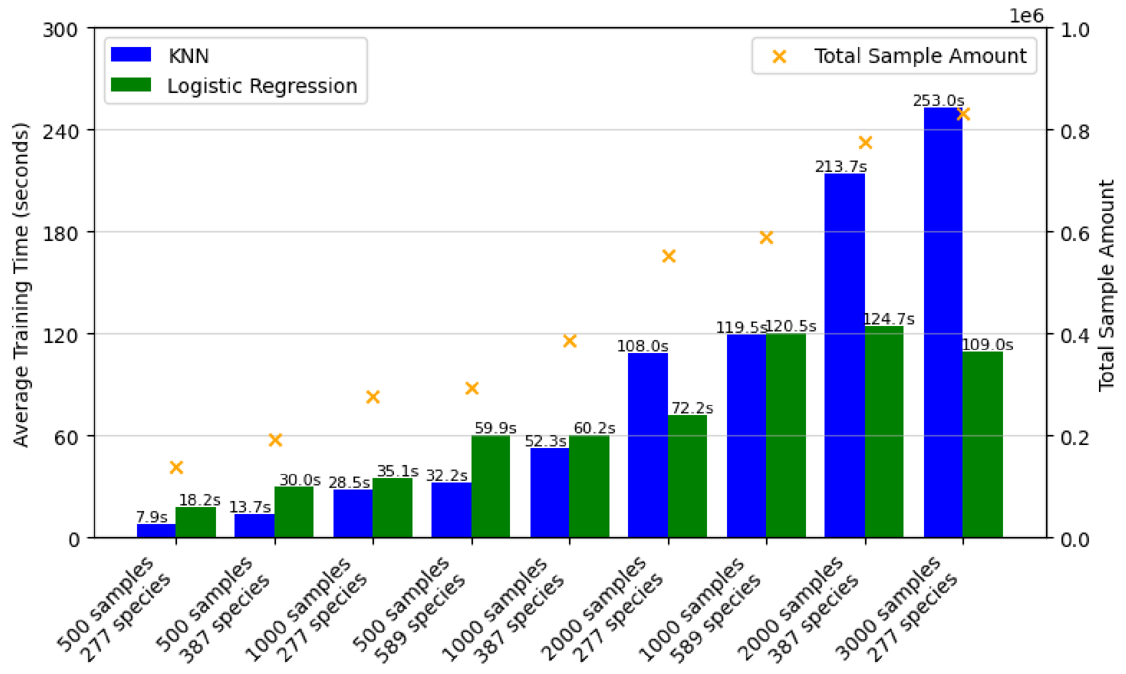


Figure 13: Training Times for a single run of KNN and Logistic Regression averaged over 5 folds and 2 dataset types.

Each bar in Figure 13 represents a training time averaged over 5 folds for each of the two dataset types. Therefore one bar is depicted for each classifier and dataset variant.

It is important to interpret these values with caution, as the workstation was shared with other users, possibly leading to resource contention during training. Additionally, software choices and hardware specifications impact the training times strongly.

While the absolute training time values may be less meaningful, the relative differences reveal a clear trend: training time increases as the sample size grows. Notably, LR scales more efficiently than the KNN classifier when handling larger datasets in terms of training time.

LR may also be influenced by the number of species in the dataset. This is suggested by the relatively constant training times observed across dataset variants $(500_{\text{samples}} \times 589_{\text{species}})$, $(1,000_{\text{samples}} \times 387_{\text{species}})$, and $(2,000_{\text{samples}} \times 277_{\text{species}})$, as well as between $(1,000_{\text{samples}} \times 589_{\text{species}})$, $(2,000_{\text{samples}} \times 387_{\text{species}})$, and $(3,000_{\text{samples}} \times 277_{\text{species}})$. This pattern suggests that a decrease in species count, accompanied by an increase in the total number of samples, does not significantly impact logistic regression training time, even though the total number of samples increases.

On the other hand, KNN training times appear to be primarily driven by the total number of samples, regardless of species count. The increases in KNN's training time between different dataset variants closely follow the increases in total sample size.

6.1.3 Effect of Number of Species on Accuracy

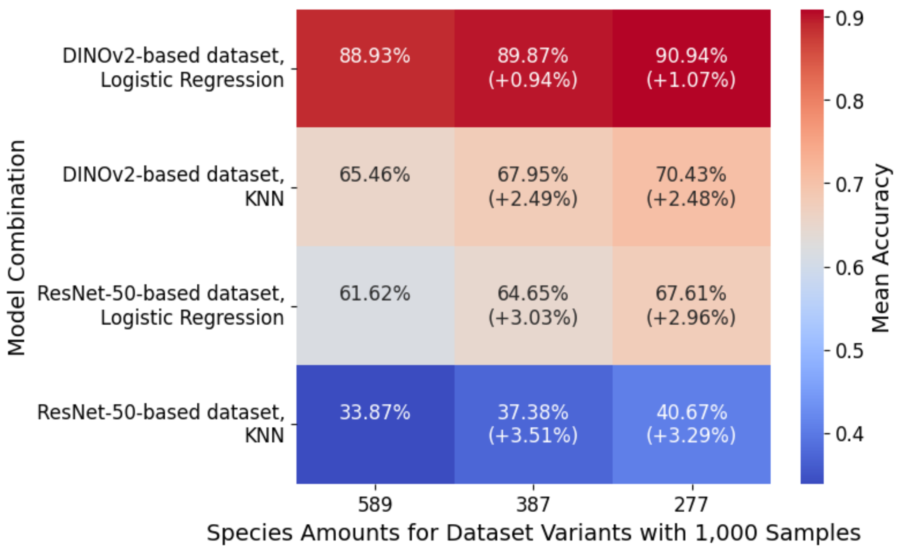


Figure 14: Heatmap showing mean accuracies and their differences between FM-CLF combinations and different amounts of species, represented by 1,000 samples each.

A clear trend emerges across all classifier and model combinations, visible in Figure 14: classification accuracy improves as the number of species decreases. This is expected, as a lower species count reduces the risk of misclassification. However, models that already achieve high accuracy show diminishing returns as the species count decreases. This pattern is particularly evident in the transition from 589 to 387 species, where the largest accuracy gain (3.51%) occurs for KNN trained on ResNet-50 embeddings, raising its total accuracy to 37.38%. In contrast, Logistic Regression trained on DINOv2 embeddings — consistently the best-performing combination — only improves by 0.94% over the same reduction in species.

Notably, unlike other FM-CLF combinations that exhibit diminishing accuracy improvements, the combination of DINOv2 embeddings and Logistic Regression shows an increasing rate of improvement as species counts decrease. Specifically, while the accuracy gain from 589 to 387 species is 0.94%, the improvement from 387 to 277 species is slightly higher at 1.07%. This is particularly interesting given the difference in species count between these reductions: 202 species between 589 and 387, compared to just 110 species between 387 and 277.

The impact of species amounts is also reflected in the Logistic Regression's loss and accuracy curves, provided in the appendix (Figure 33).

6.1.4 Effect of Sample Size per Species on Accuracy

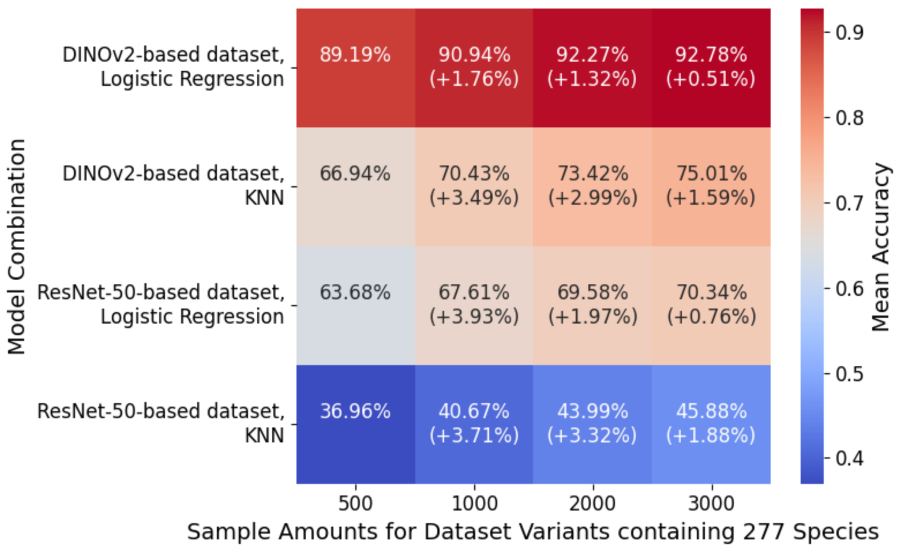


Figure 15: Heatmap showing mean accuracies and differences between FM-CLF combinations and different amounts of samples for 277 species.

Figure 15 reveals a consistent trend: increasing the number of samples per species leads to improved classification accuracy across all FM-CLF combinations. Meanwhile, similar to the effect observed when reducing the number of species, increasing the sample size per species has a more pronounced impact on FM-CLF combinations with lower initial accuracy.

The most significant accuracy gains occur between 500 and 1,000 samples, despite this step representing the smallest absolute increase in sample size (500 additional samples per species). Meanwhile, relatively seen this step means a $\times 2$ in sample amounts, which justifies large changes in accuracy.

The rate of improvement diminishes as dataset size increases, particularly for configurations that already achieve high accuracy. This trend is especially apparent for the ResNet-50-LR combination, where accuracy gains decrease from 3.93% (500 to 1,000 samples) to 1.97% (1,000 to 2,000) and further to 0.76% (2,000 to 3,000). A similar pattern is observed for DINOv2-LR, with improvements dropping from 1.76% to 1.32% to 0.51%.

On the other hand, both DINOv2-KNN and ResNet-50-KNN combinations exhibit substantial improvements over the first two sample increases (500 to 1,000 and 1,000 to 2,000), but the gains diminish significantly in the final step from 2,000 to 3,000 samples.

The influence of sample amounts is also evident in the Logistic Regression's loss and accuracy curves, as shown in the appendix (Figure 34).

6.2 Other Accuracies

From this point on, results are based on a followup training cycle using the holdout method with a 80/20 train/test split. Since the performance appears to be consistent across different dataset splits – as seen in the cross-validation results analysis (Section 6.1) – the analyses is expected to be representative for any possible dataset split.

Furthermore, to concentrate on the impact of rising sample and species amounts, **three key dataset variations** will be analyzed in greater detail:

- $277_{\text{species}} \times 1,000_{\text{samples}}$ – used as a reference point, referred to as **reference dataset**.
- $277_{\text{species}} \times 3,000_{\text{samples}}$ – to evaluate the effect of increasing the number of samples while keeping the number of species constant, referred to as **sample dataset**.
- $589_{\text{species}} \times 1,000_{\text{samples}}$ – to assess the impact of increasing the number of species while maintaining the same number of samples per species, referred to as **species dataset**.

For the sake of comparability with the other accuracy types, the top-1 (or species-level) classification accuracies for these hold-out runs are provided in Table 7.

Species Amount	**Sample Amount**	**DINOv2 LR**	**DINOv2 KNN**	**ResNet-50 LR**	**ResNet-50 KNN**
277	**3,000**	92.80	75.02	70.29	45.78
277	**1,000**	91.12	70.49	67.74	40.69
589	**1,000**	88.70	65.39	61.43	33.90

Table 7: Top-1 (species-level) classification accuracies (%) for key dataset variants and FM-CLF combinations.

Species Amount	**Sample Amount**	**DINOv2 LR**	**DINOv2 KNN**	**ResNet-50 LR**	**ResNet-50 KNN**
277	**3,000**	98.31	91.01	89.36	71.88
277	**1,000**	97.76	88.49	87.90	67.31
589	**1,000**	96.89	84.52	83.36	58.29

Table 8: Top-5 classification accuracies (%) for key dataset variants and FM-CLF combinations.

Species Amount	**Sample Amount**	**DINOv2 LR**	**DINOv2 KNN**	**ResNet-50 LR**	**ResNet-50 KNN**
277	**3,000**	94.16	78.03	72.85	49.27
277	**1,000**	92.64	73.88	70.43	44.25
589	**1,000**	91.39	70.70	66.00	39.41

Table 9: Genus-level classification accuracies (%) for key dataset variants and FM-CLF combinations.

Species Amount	**Sample Amount**	**DINOv2 LR**	**DINOv2 KNN**	**ResNet-50 LR**	**ResNet-50 KNN**
277	**3,000**	97.67	91.15	85.68	70.24
277	**1,000**	96.99	89.14	84.38	66.49
589	**1,000**	96.69	88.46	82.14	64.04

Table 10: Family-level classification accuracies (%) for key dataset variants and FM-CLF combinations.

Accuracy Comparison The trend observed in Section 6.1 repeats itself for all calculated accuracies: rising accuracies with increasing sample sizes and decreasing species amounts.

However further accuracy types were calculated, such as top-5 accuracy, which treats a sample as correct as long as the true label is within the 5 highest confidences for its prediction. Presented in Table 8, as expected the top-5 accuracies are significantly higher than top-1 accuracies, while increases are more pronounced for combinations exhibiting poor top-1 accuracies, such as the ResNet-50-KNN combination.

The same trend shows itself in genus- and family-level accuracy. Genus-level accuracy (Table 9) is the accuracy, where a sample is considered correctly predicted, if the predicted species belongs to the same genus as the true species. In a similar fashion the family-level accuracy (Table 10) indicates, that the predicted species belongs to the same family as the true species. As expected, accuracies on family-level classification are higher than on genus-level, since the family represents the broader group than the genus, which is placed lower in the taxonomic hierarchy. These accuracies and their implications will be discussed in more detail in Section 7.

6.3 Confidence Analysis

For further analysis the probabilities of the classifiers KNN (**proba function**) and LR (**softmax function**) were extracted during the same training cycle mentioned in Section 6.2.

Figure 16 shows the distribution of the top-1 confidence, which is the confidence the predicted label was chosen with. While correct and incorrect predictions were separated, confidences from all datasets were taken together and grouped after the four FM-CLF combinations, as the top-1 confidence distribution only varies marginally in between different dataset variants.

While these differences between dataset variants are reflected in the mean confidences depicted in Figure 17 and Figure 18, their variations are given additionally as box plots for the three key dataset variations in the appendix and are described in the following:

Comparing top-1 confidences between the three key dataset variations across FM-CLF combinations exhibits distinct trends for LR and KNN. Confidences for LR stay mostly the same during a sample amount increase and drop slightly for the **species dataset** in comparison to the **reference dataset**, smallering the gap between correct and incorrect predictions. KNN shows a trend, according to which confidences for correct and incorrect predictions appear to be increasing for the **sample dataset** and decreasing for the **species dataset**, both compared to the **reference dataset**. This appears to be the case for both correct and incorrect predictions. The box plots can be accessed in the appendix for DINOv2-LR (Figure 35), DINOv2-KNN (Figure 36), ResNet-50-LR (Figure 37) and ResNet-50-KNN (Figure 38).

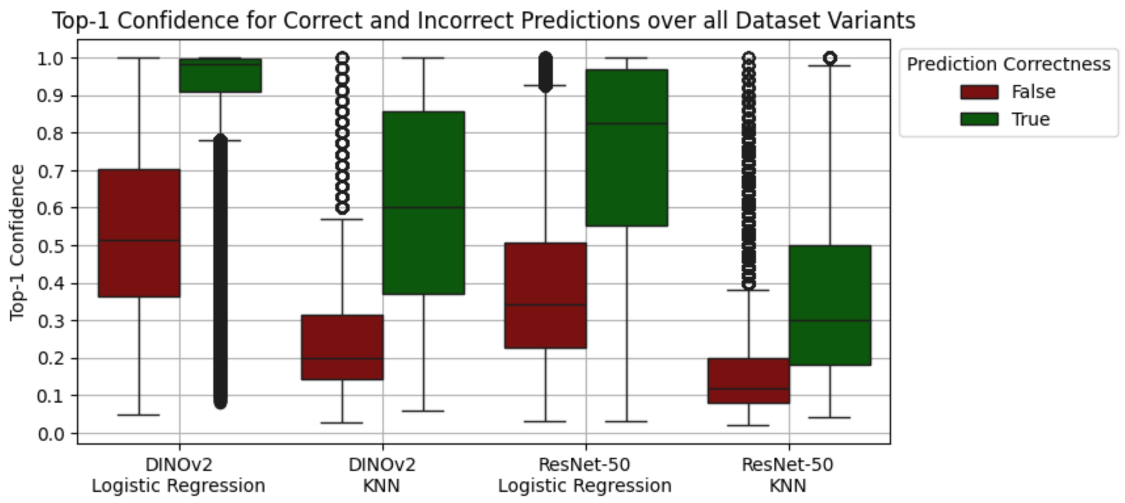


Figure 16: Top-1 confidence values for each FM-CLF combination. Each correct-incorrect boxplot pair represents nine runs over the different dataset variants.

In the following Figure 16 will be analyzed for the different FM-CLF combinations.

DINOv2-LR For incorrect predictions, the confidence values are widely spread between approximately 5% and 100%. The median confidence is slightly above 50%, indicating an almost uniform distribution of confidence levels for misclassified samples. In contrast, correct predictions exhibit extremely high confidence, with a median close to 100% and the smallest range seen in the Figure ranging from slightly below 80% to 100% confidence. The high median suggests that at least half of all correct predictions have nearly maximum confidence. The presence of numerous outliers suggests that some correct predictions still receive relatively low confidence scores.

DINOv2-KNN Incorrect predictions show a broad distribution of confidence values, ranging from close to 0% to around closely below 60%, with a median confidence around 20%, concentrating values rather in the lower half. In contrast, correct predictions exhibit a much higher confidence range, with a median confidence close to 60%. However, the distribution remains broad, meaning that even correct classifications sometimes receive relatively low confidence scores.

ResNet-50-LR Incorrect predictions cover a broad confidence range from near 0% to little over 90%, with a median sitting slightly under 35%. This suggests that half of incorrect classifications receive confidences below 35, though a the upper half spreads towards high confidence values. While confidences for correct predictions spread between close to 0% and 100%, its median is located close to 80%, indicating a strong concentration towards high confidences for correct predictions.

ResNet-50-KNN Incorrect predictions exhibit the lowest overall confidence among all FM-CLF combinations, with a median just above 10%. The half exhibiting lower confidences is much more concentrated as the median is very low as well and the upper half reaches up to shortly under 40%. Correct predictions show a much broader spread of confidences between closely above 0% to closely below 100% while the median of roughly 30% reveals a higher concentration in the lower half. This suggests that this model has the weakest separation between correct and incorrect classifications in terms of confidence values.

6.3.1 Confidence of correct Predictions across Dataset Variants

Figure 17 illustrates the mean top-1 confidence for correct predictions across all foundation model–classifier combinations and dataset variants.

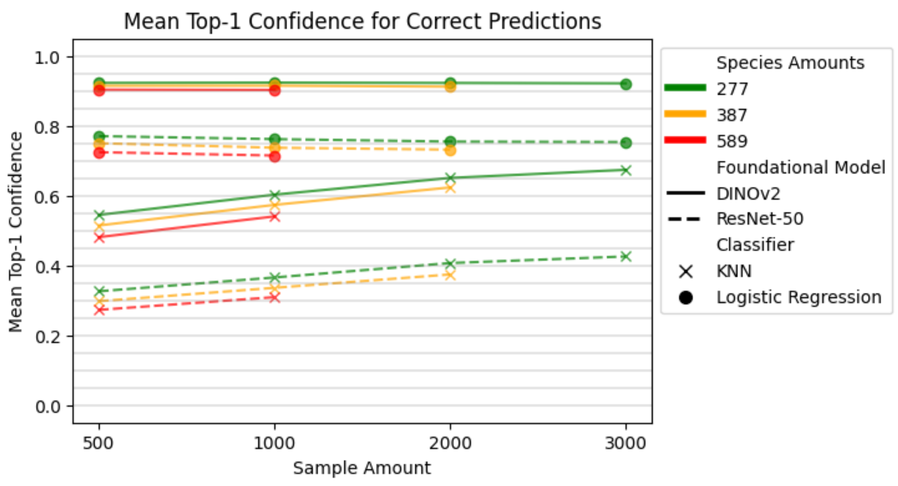


Figure 17: Average of top-1 confidence values of correct predictions for all dataset variants and FM-CLF combinations.

A clear trend can be observed as the number of species increases, mean confidence decreases in all cases. However, for the DINOv2–LR combination, confidence values across species amounts remain much closer together compared to other combinations. Additionally LR – consistently achieving the highest confidence for correct predictions – shows almost no change in confidence with increasing sample size with even a very slight decrease visible for the ResNet-50-based datasets. In contrast, for both DINOv2-KNN and ResNet-50-KNN combinations, confidence increases by approximately 10% as sample size grows from 500 to 3,000 for the dataset variants containing 277 species.

6.3.2 Confidence of incorrect Predictions across Dataset Variants

Conversely, Figure 18 presents the mean top-1 confidence for incorrect predictions, revealing similar patterns at lower confidence levels. Across all FM-CLF combinations, an increase in species count corresponds to a decrease in mean confidence, mirroring the trend observed for correct predictions. LR consistently yields higher confidence values than KNN, with confidence decreasing as species count increases. In contrast, for KNN, mean confidence rises with increasing sample size, though the effect is less pronounced than for correct predictions —- approximately a 5% increase between 500 and 3,000 samples.

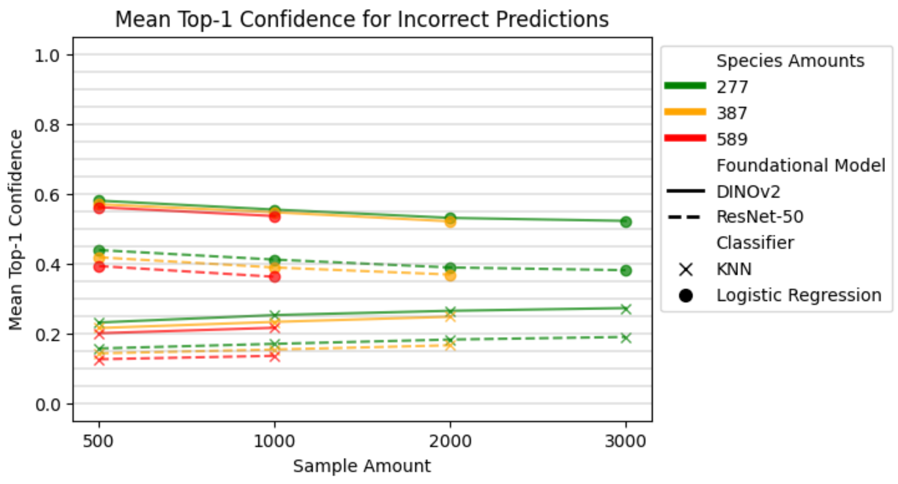


Figure 18: Average top-1 confidences of incorrect predictions for all dataset variants and FM-CLF combinations.

6.4 Species-Level Analysis

The same data as analyzed in Section 6.2 and 6.3 was used for this analysis.

To investigate class-wise accuracies and understand why certain classes perform better than others, the DINOv2-LR combination will be analyzed and then compared to all other combinations. The primary goal is to identify issues within the dataset rather than the model itself. Since DINOv2 is known for its strong generalization capabilities, any observed performance issues are more likely due to characteristics of the species rather than deficiencies in the model. Therefore, the results from Logistic Regression trained on DINOv2-based dataset variants (DINOv2-LR) served as a reference point (referred to as reference combination) for the comparison.

To assess the impact of different dataset variants on class accuracies, only the three key dataset variants were compared, as introduced in Section 6.2: **reference dataset**, **sample dataset** and **species dataset**.

Introduction to the Figure As mentioned earlier, the order *Lepidoptera* consists of different taxonomic groups, such as genus. Species sharing a common genus pose the risk of misclassification due to visual similarities. Those genus clusters are grouped together by cluster sizes, to assess its impact on the performance. Its important to note, that species that don't share a common genus can exhibit very similar appearances as well.

To analyze the impact of genera on classification performance, Figure 19 visualizes each species according to its classification accuracy within a specific training setup (DINOv2-LR trained on the reference dataset). Each marker represents a species, with its accuracy on the x-axis and its cluster size on the y-axis, where cluster size refers to the number of species within the dataset that share the same genus. Figure 20 shows the same for the **sample dataset** while Figure 21 presents the class distribution for the **species dataset**. For the other FM-CLF combinations the figures are provided in the appendix. The major differences between them and the reference combination (DINOv2-LR) are analyzed below.

While each marker represents a specific class, the colors categorize them into different groups:

- **Red, yellow, and green** marker (referred to as traffic light colors) indicate classes in the lower (red), middle (yellow), and upper (green) thirds of class accuracy. These accuracy categoies are computed for each cluster size individually and are specific to this the reference combination (DINOv2-LR) on the reference dataset (277 species with 1,000 samples each).
- **Blue and cyan** marker represent species that are *Spec.*-labels (blue) or belong to a *Spec.*-cluster (cyan)
- **Pink** marker represent specific species which were specifically selected due to their strong variations in class accuracy in between different dataset variants. The numbers refer to the specific label used in the source code for reference.

All colors for each specific class remain consistent across all figures, ensuring comparability between different FM-CLF combinations and dataset variants, meaning that if class X is represented by a yellow marker in the reference dataset for the reference combination, it is represented by a yellow marker in all other figures as well.

6.4.1 Analysis of Label Distribution and Cluster Size

As the number of species increases, the cluster sizes (CS) grow accordingly. For example, the (*Eupithecia*) cluster expands from a CS of 2 in the reference dataset to a CS of 10 in the species dataset. Similarly, the *Idaea* cluster increases from a CS of 7 to 11. These two clusters become the largest in the dataset. On top, both of them contain a general *Spec.*-label and will be discussed further in Section 6.4.3.

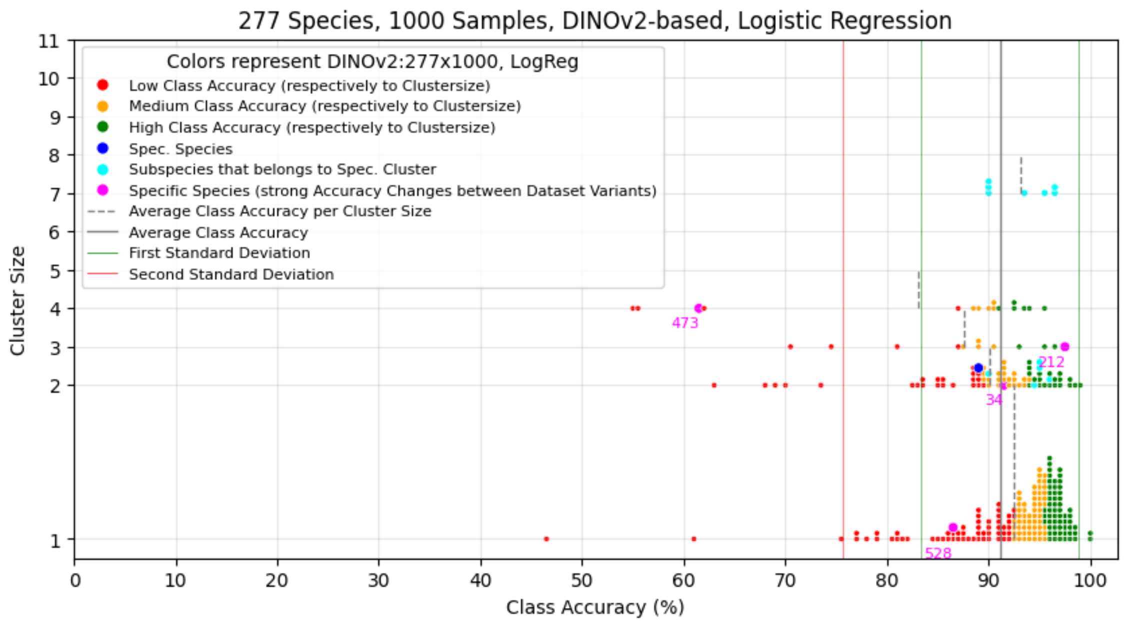


Figure 19: Class accuracy distribution for the DINOv2-LR combination trained on the reference dataset.

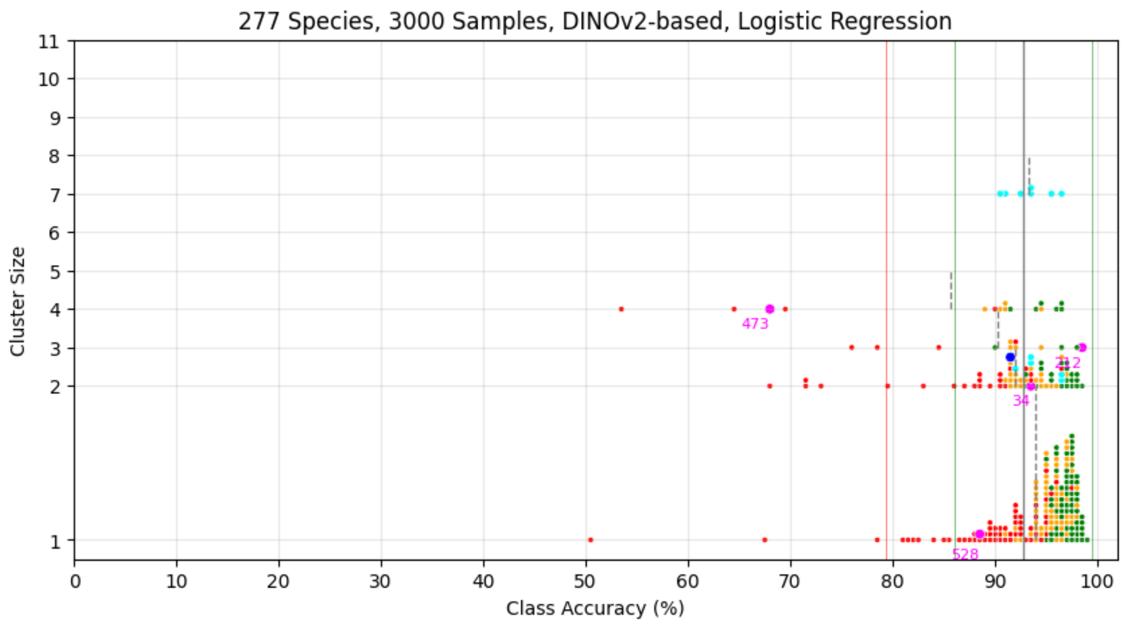


Figure 20: Class accuracy distribution for the DINOv2-LR combination trained on the sample dataset.

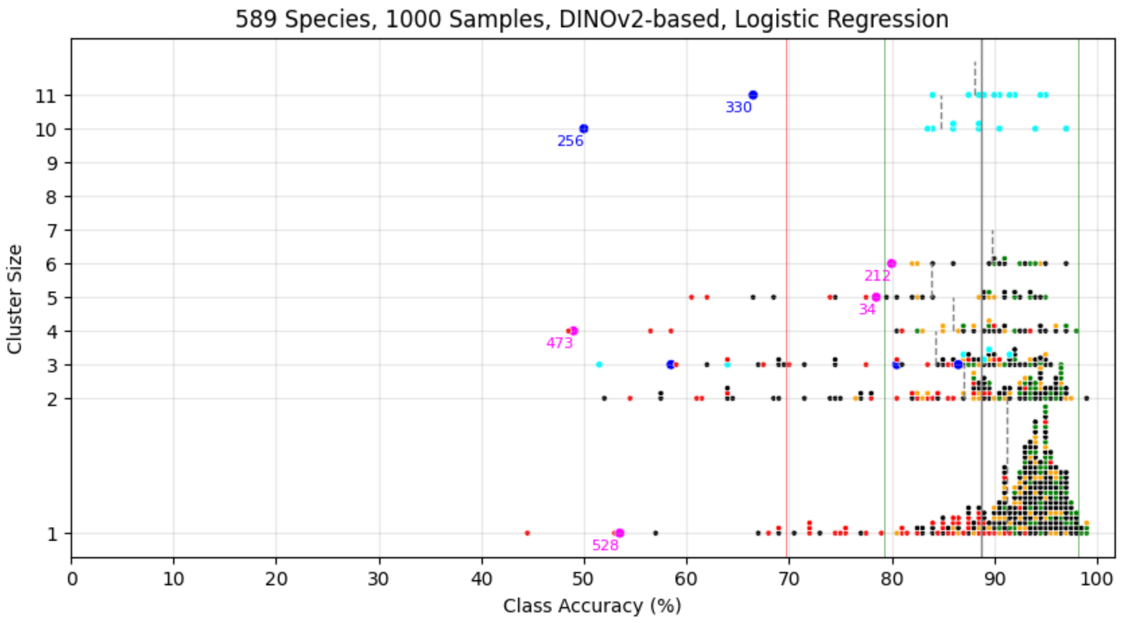


Figure 21: Class accuracy distribution for the DINOv2-LR combination trained on the species dataset.

When examining all three figures together, a slight mixing of traffic light colors is evident in the middle accuracy range, where green and red marker blend with yellow for the sample and species dataset. Nonetheless, green marker predominantly occupy the upper half of the accuracy range, while red marker are mostly found in the lower half across all dataset variants. Moreover, the classes are relatively densely grouped. For instance, the **reference dataset** shows a mean class accuracy of 91.1% with a standard deviation of 7.7% (with only marginal differences across dataset configurations).

In the **reference dataset**, the difference between mean accuracy across cluster sizes (as indicated by the dashed lines) is no more than 9.4%. For example, species not assigned to any cluster (CS of 1) have a mean accuracy of 92.4%, while a CS of 4 has a mean accuracy of 83.0%. In general, mean accuracy declines as the cluster size increases from 1 to 4. An exception is seen at CS of 7, where performance exceeds that of CS of 1. This cluster corresponds solely to the *Idaea* genus, and the classifier appears to distinguish its species (cyan markers at CS of 7 or 11) particularly well, resulting in a high mean accuracy.

As the sample size increases in the species dataset, the mean accuracy for each CS increases by 1.5% to 2.7%. Increasing the number of species on the other side also alters the composition of CS groups, as species are reallocated from lower to higher CS layers due to the introduction to more species belonging to the same genus cluster. This shift complicates direct comparisons of mean accuracies between the reference and species datasets and makes it challenging to isolate the impact of additional species on individual class accuracies. An analysis of this impact on specific species is provided in Section 6.4.2. Nevertheless, the overall trend of decreasing mean accuracy with increasing cluster size remains consistent.

DINOv2-KNN The traffic light color pattern is similar to the reference combination, although the spread of class accuracies is notably wider which reflects in the significantly higher standard deviation of 18.8% form the mean accuracy of 70.5%. Mean class accuracy further decreases as cluster size increases, with a total gap of 21.6% between a CS of 1 (73.7% mean accuracy) and a CS of 4 (52.1% mean accuracy). Notably, the *Idaea* cluster (CS of 7) maintains a high mean accuracy of 70.5%. In addition, the **sample dataset** shows an improvement in mean class accuracies for each cluster size by between 4.3% and 7.3% compared to the **reference dataset**. Graphics can be found in the appendix for the reference dataset (Figure 39), sample dataset (Figure 40) and species dataset (Figure 41).

ResNet-50-LR In this configuration, the traffic light colors are more blended than in the DINOv2-KNN case. The middle range is broader, with yellow appearing more frequently in both the lower and upper halves, although green and red still dominate their corners. Overall, the model achieves a mean class accuracy of 67.7% with a standard deviation of 13.5%, which is slightly lower in both spread and mean accuracy compared to class accuracies of the DINOv2-KNN combination.

Accuracy declines as cluster size increases: for a CS of 1, the mean accuracy is 70.6%, while for a CS of 4 it drops to 48.3%, yielding a gap of 22.3%. Interestingly, the accuracy for a CS of 3 is higher than for a CS of 2 – a trend not observed with DINOv2 embeddings. Additionally, the accuracy for the largest cluster (CS of 7) falls to 62.2%, which is notably lower than that for a CS of 1. Nevertheless, the overall downward trend in accuracy with larger cluster sizes remains. Compared to the **reference dataset**, the **sample dataset** exhibits an improvement in mean class accuracies for each cluster size by between 2.5% and 3.9%. For further insight please refer to Figure 42 for the **reference dataset**, Figure 43 for the sample dataset and Figure 44 for the species dataset, which are all provided in the appendix.

ResNet-50-KNN This combination is the lowest performing among those evaluated which reflects in the average class accuracy of 40.7% for the **reference dataset** with a standard deviation of 18.4%, which is similar compared to that observed for the DINOv2-KNN combination. Furthermore the mixture of traffic light colors is the strongest with single instances of green and red marker being found at the opposite sides. The accuracy gap across cluster sizes is 21.8%: species in a CS of 1 achieve 43.4% accuracy, whereas those in a CS of 4 drop to 21.6%. The accuracy for a CS of 3 is higher than for a CS of 2, and the accuracy for the largest cluster (CS of 7) further declines to 28.7%. This pattern mirrors the trend seen for ResNet-50-LR. Moreover, increasing the sample size leads to an improvement in mean accuracy per cluster size by between 3.9% and 5.5%. For a closer look please refer to the graphics provided in the appendix for the reference dataset (Figure 45), sample dataset (Figure 46) and species dataset (Figure 47).

6.4.2 A Closer Look at Specific Species

The following observations are based on the DINOv2-LR combination, depicted in Figures 19, 20, and 21.

The species *Speyeria aglaja* (pink marker 528), *Agriphila tristella* (pink marker 34), *Eilema complana* (pink marker 212) and *Pieris mannii* (pink marker 473) were selected for their significant variations in class accuracy across different dataset variants while belonging to different cluster sizes. The number for each species represents the actual label used in the classification process.

It is important to note that when discussing species that were commonly confused with a given species, only those that were misclassified more than once are discussed, as a single misclassification does not indicate a pattern.

- **Species *Speyeria aglaja* (528), CS of 1 in the reference dataset**

  Species 528 benefits from an increased number of training samples, with its accuracy improving from 86.5% in the **reference dataset** by 2% in the **sample dataset**. However, when the total number of species is expanded, its accuracy drops by 33% compared to the **reference dataset**. This decline is accompanied by a significant increase in the number of different species it is commonly confused with, rising from 5 in the **reference dataset** to 12 in the **species dataset**. The two labels label 528 was mostly confused with are provided in Figure 22 and are both introduced in the **species dataset**.

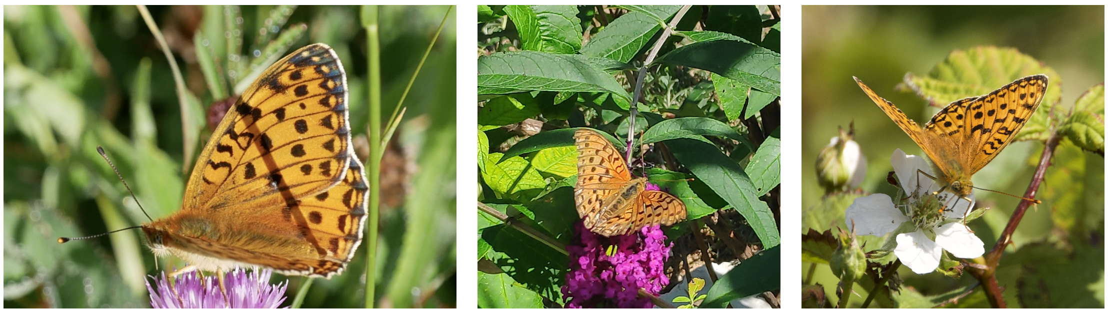
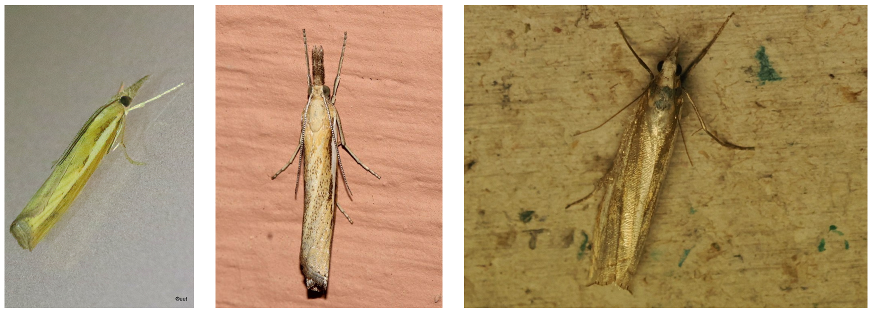

(a) 528(sam, l) (b) 271(sam, e) (c) 272(sam, f)

Figure 22: Examples of visually similar species: (a) *Speyeria aglaja* (528), correctly predicted in 107 out of 200 cases; (b) *Fabriciana adippe* (271), wrongly predicted 30 times; (c) *Fabriciana niobe* (272), wrongly predicted 16 times.

- **Species *Agriphila tristella* (34), CS of 2 in the reference dataset**

  Species 34 begins with a high accuracy of 91.5% in the reference dataset, which improves by 2% with the introduction of more samples. However, when the number of species is expanded from 277 to 589, accuracy decreases by 13%, and its cluster size increases by 3. Additionally, the number of species it is misclassified as rises from 2 in the **reference dataset** to 6 in the **species dataset**.

  Among the 39 misclassified instances (as one of the 6) in the **species dataset**, 24 occurred within the same genus cluster, while 15 involved species outside this cluster. The two species it was most often confused with are shown in Figure 23 while neither of which appear in the **reference dataset** but only in the **species dataset**.



(a) 34(sam, h) (b) 32(sam, g) (c) 166(sam, a)

Figure 23: Examples of visually similar species: (a) *Agriphila tristella* (34), correctly predicted in 157 out of 200 cases; (b) *Agriphila selasella* (32), a species within the same genus cluster, misclassified 17 times; (c) *Crambidae indet.* (166), an unrelated species outside any cluster, misclassified 7 times.

- **Species *Eilema complana* (212), CS of 3 in the reference dataset**

  Species 212 starts with a very high accuracy of 97.5% in the reference dataset, improving by an additional 1% in the **sample dataset**. However, when the number of species increases, its accuracy drops significantly by 17.5% compared to the reference dataset.

  In the reference dataset, misclassifications are rare and dispersed across multiple species, each misclassification occurring for a different label. In contrast, in the **species dataset**, all misclassifications are concentrated within its genus cluster, specifically with species 211 and 215 which are illustrated in Figure 24 and both only appear to be in the **species dataset**.

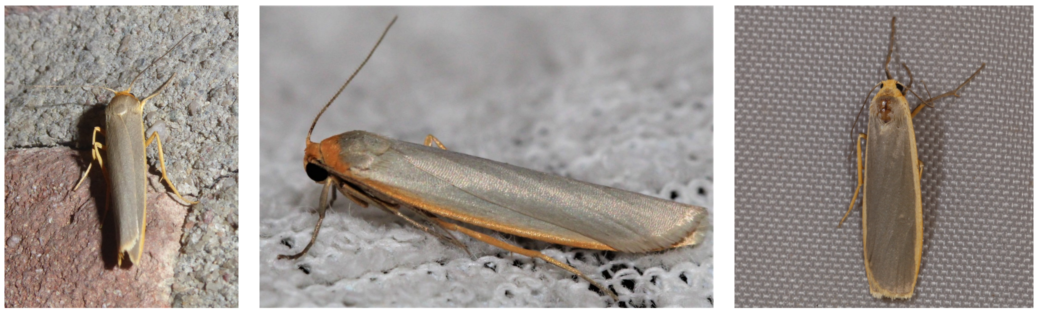

(a) 212(sam, c) (b) 211(sam, b) (c) 215(sam, d)

Figure 24: Examples of visually similar species: (a) *Eilema complana* (212), correctly classified in 160 out of 200 cases; (b) *Eilema caniola* (211), wrongly predicted in 28 cases; (c) *Eilema lurideola* (215), wrongly predicted in 8 cases.

- **Species *Pieris mannii* (473), CS of 4 in the reference dataset**

  Species 473 starts with an accuracy of 61.5% in the reference dataset. In the **sample dataset**, its accuracy improves by 6.5%, but when the number of species increases, it drops by 12.5% compared to the **reference dataset**. The number of species it is confused with rises from 5 in the **reference dataset** to 9 in the **species dataset**.

  A total of 96 instances were misclassified, with 81 of them occurring within the same cluster. The two species it was most confused with are shown in Figure 25 and do occur in the reference and **species dataset**. Even though the cluster size did not increase for the **species dataset** the amount of misclassifications within the cluster increased from 64 in the reference dataset to 81 in the **species dataset**.

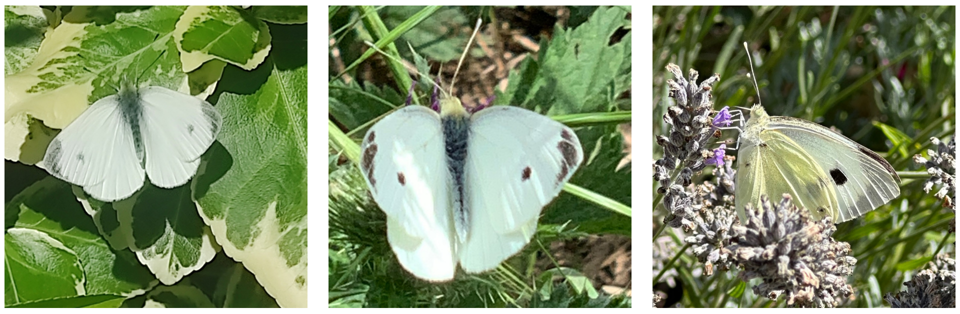

(a) 473(sam, j) (b) 475(sam, k) (c) 472(sam, i)

Figure 25: Examples of visually similar species: (a) *Pieris mannii* (473), correctly predicted in 98 out of 200 cases; (b) *Pieris rapae* (475), wrongly predicted in 41 cases; (c) *Pieris brassicae* (472), wrongly predicted 23 times.

Confusion matrices for the different genus clusters can be found in the appendix for the **reference dataset** and the **species dataset**. Figure 48 and 49 show predictions for the *Agriphila* genus (to which label `34` belongs), Figure 50 and 51 show predictions for the *Eilema* genus (part of which is label `212`) and Figure 52 and 53 show predictions for the *Pieris* genus (of which label `473` is a species).

Table 11 presents the classification results for the discussed species, detailing the two most frequent misclassifications for each species, along with the number of instances they were predicted and the average confidence scores. The confidence scores describe how certain the classifier classified the predicted labels on average. The highest confidence values consistently correspond to correct classifications, while the difference to the second most predicted species' mean confidence varies between 12.29% (for label `473`) and 16.32% (for label `34`). The differences in confidence levels indicate varying degrees of certainty in the model's predictions, with some misclassifications occurring at relatively high confidence levels.

True Label	Predicted Label	Instances	Average Confidence (%)
528	**528**	**107**	**66.63**
	271	30	46.21
	272	16	52.23
34	**34**	**157**	**77.53**
	32	17	61.21
	166	7	49.78
212	**212**	**160**	**88.50**
	211	28	72.35
	215	8	51.20
473	**473**	**98**	**70.76**
	475	41	58.47
	472	23	58.33

Table 11: Classification results for selected species, showing their two most common wrong predictions, the number of instances they were classified and the average confidence with which the prediction was carried out. This table shows results of the DINOv2-LR combination trained on the **species dataset**. The test-split consisted of 200 samples for each species.

6.4.3 Impact of Spec Label on its cluster

Spec. Clusters Results for the DINOv2-LR combination, Figure 19 trained on the **reference dataset**, shows a distinct cluster of size 7, corresponding to the *Idaea* (`330`) genus (marked in cyan). Notably, this cluster lacks the *Idaea Spec.* label (which would be represented as a blue marker) in this dataset variant. However, upon increasing the species count to 589, this genus cluster expands from 7 to 11 labels. The second-largest cluster, increasing from 2 to 10 labels, belongs to the

Eupithecia genus and introduces the *Eupithecia Spec.* (256) label when increasing the species amount.

Despite their large cluster sizes, these species exhibit relatively high classification accuracies. However, the low accuracy of *Idaea Spec.* (330) and *Eupithecia Spec.* (256) (observable as blue marker for cluster sizes 11 and 10, respectively) suggests that the model tends to classify general species (which were not further categorized into species by experts) into specific species within the same genus. Conversely, species occasionally misclassify samples from the *Spec.* label as their own, though this occurs less frequently than the former case.

Confusion matrices for both genera and dataset variants (reference and species dataset) verify that suspicion and are provided in the appendix: Figure 56 and 57 for the *Eupithecia* genus and Figure 54 and 55 for the *Idaea* genus.

6.4.4 Inter Cluster Confusions

While species within a genus cluster pose high risk of confusion, species of different genera can also exhibit similar visual features. This becomes apparent when taking a look at Figure 26, where the corners of the white squares show genera which were confused in that run. The species of the *Bologira* genus for instance were often confused with species belonging to the *Melitaea* genus and vice versa. The same appears to be the case for the genera *Agriphila* and *Crambus*. The clearly visible diagonal, representing misclassification within genus clusters is evidence of genera posing a challenge for fine-grained image classification.

For all other FM-CLF combinations – which will be discussed in the following – the confusion matrices for the **species dataset** are provided in the appendix: Figure 58 for DINOv2-KNN, Figure 59 for ResNet-50-LR, and Figure 60 for ResNet-50-KNN.

Findings in other FM-CLFs' Confusion Matrices While the overall pattern observed in Figure 26 remains consistent across the other three FM-CLF combinations, additional trends emerge. Notably, there is frequent confusion between species of the genera *Agrochola* and *Conistra*. Furthermore, some misclassifications occur asymmetrically: for instance, species from the *Scopula* genus are often mistaken for *Idaea*, and *Idaea* species frequently misclassified as *Eupithecia*, yet the reverse confusion (*Eupithecia* to *Idaea* or *Idaea* to *Scopula*) is not observed in the same strength, as indicated by the absence of a clear corresponding patch in the confusion matrices.

On a broader level, the ResNet-50-KNN combination exhibits a nearly uniform distribution of misclassifications across all species, suggesting a higher degree of unrelated confusions, visible as noise. However, ResNet-50-LR and DINOv2-KNN show a more structured pattern, with less dispersed prediction errors.

This concludes the section presenting the experiments' results and their analysis, paving the way for further discussion in the following section.

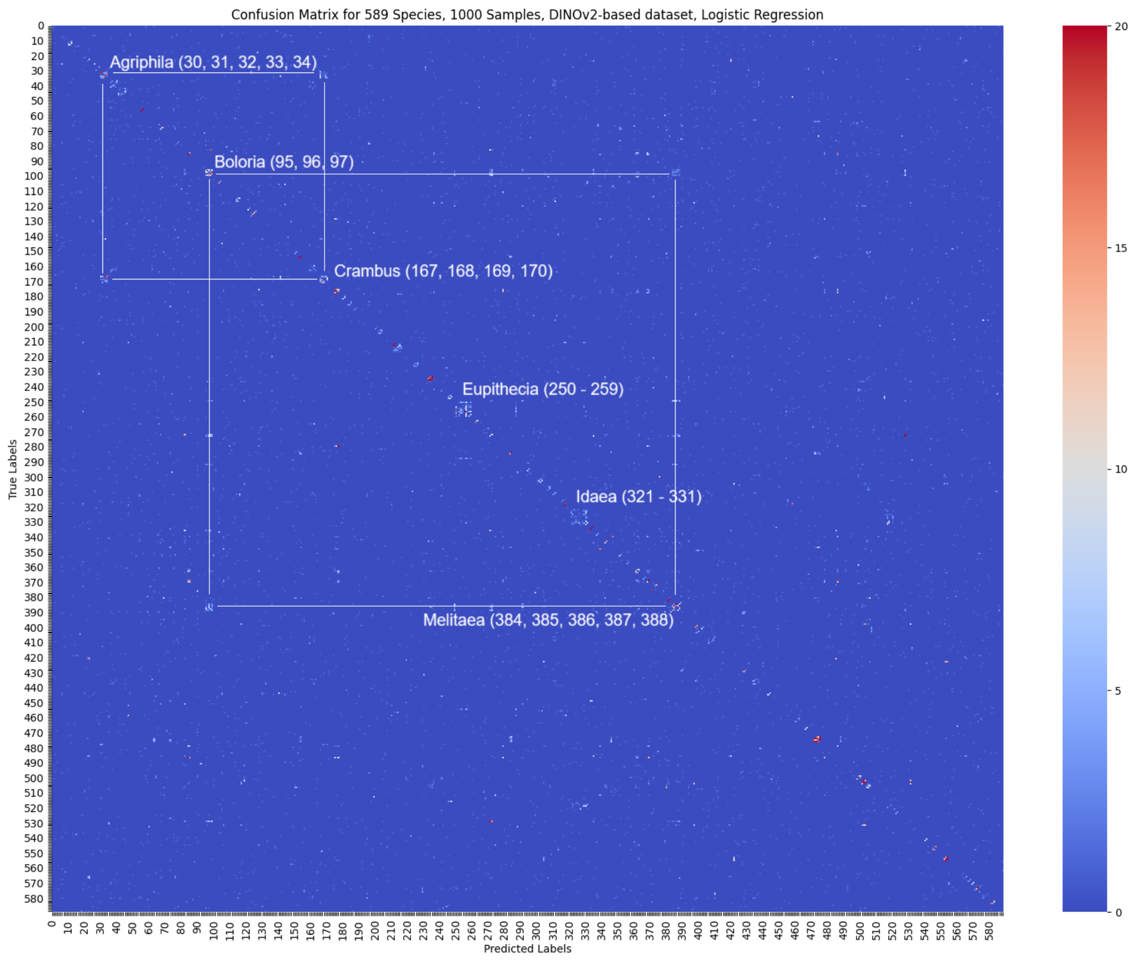


Figure 26: Confusion matrix including all 589 species (sorted after name and therefore grouped by genus) of the **species dataset** for the DINOv2-LR combination. With correct classification numbers being set to zero, the focus lies on the misclassifications. The corners of the squares indicate which genus clusters are confused with each other.

7 Discussion

7.1 Findings and Implications

In the following the results and analyses from Section 6 will be discussed and interpreted.

7.1.1 Cross-Validation Results

Returning to Section 6.1, consistent trends could be observed in Figure 12 which were derived using cross-validation, underscoring the reliability of the findings. Specifically, the stable performance gaps between different classifiers and the consistent in- and decrease in accuracy across dataset variants for each combination reflect the underlying characteristics of the feature embeddings.

Performance of Foundation Models A key observation is, that classifiers trained on DINOv2 embeddings consistently outperformed those trained on ResNet-50 embeddings. This suggests that DINOv2 produces a more robust and informative feature space for the fine-grained classification task, detecting differences between different species with more precision than ResNet-50.

DINOv2 uses self-supervised learning to extract high-level, invariant representations that capture semantic information beyond pixel-level details, which makes it particularly effective at capturing complex patterns, further enhancing its generalizability on diverse datasets (Oquab et al., 2024). In contrast, ResNet-50 – trained on the ImageNet1k dataset using supervised learning – is biased toward dataset-specific features. If ResNet-50 exhibits a bias toward texture rather than shape (Geirhos et al., 2018), that could have significant negative impact the ability to distinguish between species. In theory, this could also benefit classification, as the shapes can differ strongly while the texture stays consistent within the same species (e.g. due to different angles for pictures of the same species). Especially if certain species exhibit very characteristic textures, essential to recognize for successful classification, this could be an advantage. However this claim is not supported by any results. On the other side, texture heavily depends on the factors like brightness, sharpness, and color, which vary significantly when using citizen science data.

By contrast, DINOv2's self-supervised training (on approximately 110 times the data that was used for training ResNet-50), demonstrates superior generalization, which is supported by excellent results, as discussed further on. Given the complexity of the classification task, ResNet-50 still achieved a reasonable level of generalization.

Furthermore, low standard deviations across cross-validation splits indicate that both foundation models' embeddings generalized well, with a minimal impact of dataset partitioning on the classifiers' performances.

Performance of Classifiers Logistic Regression (LR) consistently outperformed K-Nearest Neighbors (KNN) across all configurations, despite testing a wide range of k values (1 to 1000). The minimal impact of varying k suggests that local neighborhoods in the feature space do not reliably reflect class distributions, potentially due to noise or less relevant features retained after dimensionality reduction using PCA.

While PCA causes information loss, the consistently small accuracy variations across a wide range of k values stayed persistent across multiple tested target dimensions using different reduction methods (PCA/UMAP), which indicates that KNN struggles regardless of the reduction method and its' k values. This could indicate that the dataset structure might just favor a linear decision boundary, making LR a more suitable choice, while KNN's distance-based approach becomes ineffective in a feature space with overlapping embeddings. As only k was evaluated, it is also possible that further tests using different distance metrics or other algorithms to compute the nearest neighbors could have improved its performance.

Computational Efficiency As shown in Section 6.1.2 training time for KNN increases roughly linearly with the number of training samples. LR strongly benefits from a reduction in training time when there are fewer target classes involved. As illustrated in Figure 13, LR clearly outperforms KNN in terms of training time when handling larger sample amounts, using the given configurations.

Taking a step back: while the training time for the classifiers took only minutes (using two NVIDIA RTX A5000), fine-tuning foundation models would take significantly longer. While DINOv2 is suggested to be used with linear probing (META, 2023), its already decent performance might not warrent the effort required for fine-tuning.

ResNet, on the other hand, is known to benefit significantly from fine-tuning, especially in comparison to linear probing (Kaplun et al., 2023). Given that its performance lagged behind DINOv2, ResNet-50 could be a worthier opponent if fine-tuned.

Impact of Species and Sample Amount on Accuracy Figure 14 shows that reducing the number of species leads to a slight accuracy increase, even though the improvements remain relatively small but even across FM-CLF combinations, despite strong differences in species amounts. This suggests that the foundation models produce robust, generalizable embeddings, allowing classifiers to perform well regardless of species amount. However, this observation is made based on a limited number of species using a balanced dataset and therefore the extent to which this observation is applicable remains uncertain.

Impact of Sample Amount on Accuracy Moving on to sample amounts, results shown in Figure 15 underline that increasing the sample size benefits lower-performing configurations more substantially, while higher-performing configurations approach an accuracy plateau as additional data yields less significant improvements. Notably, while improvements drop when using 3,000 samples, the increase in accuracy remains relatively high between 500 to 1,000 and 1,000 to 2,000 samples. This suggests that there may be a trade-off regarding the amount of data gathered, as close to optimal performance might be achieved with a limited number of samples per species (e.g. around 2,000), thereby reducing the need for extensive data collection.

As accuracy only increases marginally when reducing species amounts or increasing sample size, the FM-CLF combinations appear to reach a baseline performance with very limited data. Once this plateau is reached, when increasing sample size or decreasing species amounts, accuracy improves at a much slower rate.

7.1.2 Other Accuracies

While the top-1 accuracy shows that specifically ResNet-50-KNN exhibits significant deficits, the top-5 accuracy indicates that KNN trained on ResNet-50 embeddings is well able to predict a direction with relative certainty. This suggests that while the feature space provided by ResNet-50 is not sufficiently discriminative for fine-grained classification when using KNN, it still captures meaningful relationships between classes. The improved performance in top-5 accuracy indicates that the feature embeddings preserve a structured similarity, allowing KNN to identify a set of plausible candidates even if the exact match is uncertain.

Furthermore, the rather small gap between genus-level and top-1 (species-level) accuracy, with a bigger gap for the species dataset, suggests that the model is not significantly better at predicting genera than species. This is expected, as species within the same genus often exhibit strong visual similarities — if the model struggles to distinguish species, it is likely to struggle with genera as well. On the other hand, if the model classifies a moth or butterfly within its genus, it is likely to be the correct species. A reason for this behavior could lie in the variations in data quality, as some samples might be so out of focus or color-distorted, that it's very difficult to distinguish between genera, while sharp images provide sufficient detail for correct intra-genus classification. However, the larger gap in the species dataset indicates that while species-level classification remains challenging, some genus-level information is still retained. In contrast, smaller gaps in other dataset variants imply frequent misclassifications that go beyond the genus level.

Beyond the genus level: Family-level accuracies closely align with or outperform top-5 accuracies, suggesting that while species distinctions are challenging, the model effectively captures broader taxonomic relationships. This implies that misclassifications tend to occur within their families rather than across more distant taxa.

7.1.3 Confidence Analysis

Foundation Models Across all analyses, classifiers being trained on DINOv2 embeddings consistently outperform those trained on ResNet-50 embeddings in terms of producing higher confidences for correct predictions and better separation between correct and incorrect classifications. This highlights the superior feature quality of DINOv2's self-supervised learning approach, being trained on a dataset magnitudes larger than ImageNet1k.

Classifier LR outperforms KNN in both accuracy and confidence metrics across both embedding types and all dataset variants. Its ability to model linear decision boundaries effectively leverages the structure of embedding spaces created by both foundation models. In contrast, KNN struggles with low confidences due to its reliance on local neighborhood distances, which are less reliable in high-dimensional or overlapping feature spaces, as discussed before.

Dataset Variants While increasing sample size improves KNN's performance slightly by providing more reliable neighbors for distance-based decisions, LR remains largely unaffected by dataset size, possible due to its global optimization approach and datasets already providing a sufficient number of samples to perform decently. The decreasing trend in mean confidence with increasing species count reflects the growing complexity of classification tasks as class diversity increases.

7.1.4 Cluster Analysis

In the course of conducting the cluster and label accuracy analysis in Section 6.4, visualizations show that both DINOv2 and ResNet-50 tend to have similar classification strengths and weaknesses for the same species, as indicated by the similarities in traffic light color distribution (e.g. in Figure 19 and Figure 42). This suggests, that both foundation models detect similar features, while DINOv2 appears to distinguish with higher precision between species and their small differences.

Moreover, a higher number of species within a cluster appears to correlate with a slight decrease in average accuracy for those species, highlighting the risk of intra-genus misclassifications. However, as the accuracy results indicate, this risk has a smaller impact on overall performance than initially expected, as evidenced by only marginal improvements in genus-level accuracy.

Specific Species Furthermore, a detailed examination of specifically selected species for their accuracy variations between dataset variants indicates that the introduction of new species with similar visual features is a primary source of confusion, although clear distinctions in confidence between correct and incorrect predictions seem to persist, which indicates a clear separation between the feature embeddings of correctly classified samples and those of misclassified samples

Spec. Labels Finally, it is noteworthy that the error-prone *Spec.* labels performed surprisingly well. Despite these samples being technically unlabeled as specific species, the classifiers learned to recognize which species are inherently difficult to distinguish. Consequently, the DINOv2-LR combination achieved an accuracy of 50% or higher on these labels. While this means that the species labeled as *Spec.* were technically assigned to the wrong label, that does not mean that the species in that label were misclassified. Conversely, the classification itself could have been more accurate, possibly placing the species correctly within its wider genus. With a low amount of 5 *Spec.* labels that belong to a genus cluster in the species dataset, the impact of not removing those species is definitely minor, as the species within the same clusters were not regularly misclassified as the *Spec.* label.

7.2 Limitations and Challenges

The project faced several limitations and challenges that affected its overall performance. In particular, the dataset suffered from inherent bias, exacerbated by poor quality control practices that allowed unqualified and trap images – or images containing multiple insects – to slip through. In addition, *Spec.* labels were not removed beforehand, since only valid species should have been retained to maintain clarity in classification.

Due to time constraints, extensive testing with UMAP and other feature reduction techniques, as well as experimenting with different hyperparameters for KNN, was limited. Exploring alternative methods could potentially improve KNN's performance, which warrants further investigation in future experiments.

7.3 Future Work

As DINO-LR and ResNet-50-LR combinations approach a performance ceiling, it may be necessary to fine-tune the underlying foundation models to push beyond that ceiling, especially with ResNet-50. Given that DINOv2 is already performing at a strong level, the effort required for additional fine-tuning may not justify the potential gains. On the other hand, since ResNet-50's performance is significantly lower than that of DINOv2, fine-tuning ResNet-50 for fine-grained classification tasks could lead to substantial improvements, a strategy that is commonly employed, as shown in Section 2.

Another interesting pathway to improve accuracy would be, to test alternative classifiers or improve the dataset curation process. Additionally, experimenting with dataset variations, including intentionally imbalanced datasets with strongly underrepresented species, could provide further valuable insights for exploring data efficiency in this domain, leveraging the full 7,442 species contained in the initial dataset and embracing the class imbalance.

While the key challenge in fine-grained image classification lies in the task to distinguish between subtle differences, another type of information might help in the classification process. A promising direction could also lie in the incorporation of multimodal information, such as geographic location, to leverage additional contextual data for improved species identification. (Yang et al., 2022) The dataset used in the conducted experiments also initially included geographical data for each sample, providing opportunities for further exploration and potential improvements to this work.

8 Conclusion

This work highlights the challenges of fine-grained insect classification, where visual similarities within genera but also the broader family often lead to misclassifications. While in the experiments, KNN struggled with fine-grained distinctions, due to its reliance on local feature space relationships, Logistic Regression consistently outperformed it by leveraging a global decision boundary. Similarly, DINOv2 embeddings significantly surpass ResNet-50, demonstrating their effectiveness in capturing fine details without fine-tuning.

Changes in the number of species or samples only resulted in marginal performance differences for each combination of foundation model and classifier (FM-CLF). These findings indicate that the FM-CLF combinations reach a threshold, which is difficult to surpass significantly, unless that means strong increases in sample size or decreases in species amounts.

However, DINOv2-LR being the by far best FM-CLF combination, emerges as the optimal choice for fine-grained insect classification out of the experiments conducted, making it a strong candidate for real-world applications. Fine-tuning the foundation model or experimenting with alternative linear classifiers could further improve its performance, but even in its current form, the model's accuracy enables potential usability in biodiversity monitoring and conservation efforts.

Beyond this work, this approach could extend to other domains, including different insect taxa and broader nature datasets. The ability of DINOv2-LR to handle noisy, variable-quality data makes it particularly valuable for citizen science projects, where data is often collected under non-standardized conditions. By integrating such models into citizen science platforms, we could enhance large-scale biodiversity monitoring, improve species identification accuracy, and empower non-experts to contribute meaningfully to ecological research.

A Appendix: Choice of Best Parameters

Cross-validation results for Logistic Regression, averaged over DINOv2-based dataset variants

Choice	**Learning Rate**	**Epochs**	**Mean Accuracy (%) ± Std (%)**
x	0.0005	1500	**89.00** ± **1.99**
-	0.0005	2000	89.95 ± 2.07
-	0.0010	1500	89.81 ± 2.17
-	0.0010	2000	89.64 ± 2.24
-	0.0050	1500	88.87 ± 2.36
-	0.0050	2000	88.64 ± 2.39

Table 12: The table is sorted by mean accuracy, the highest mean accuracy and lowest standard deviation are marked in bold.

Cross-validation results for KNN, averaged over DINOv2-based dataset variants

Choice	**Neighbors**	**Mean Accuracy (%) ± Std (%)**
-	20	**68.59** ± 4.47
x	35	68.43 ± 4.41
-	50	68.07 ± **4.39**

Table 13: The table is sorted by mean accuracy, the highest mean accuracy and lowest standard deviation are marked in bold.

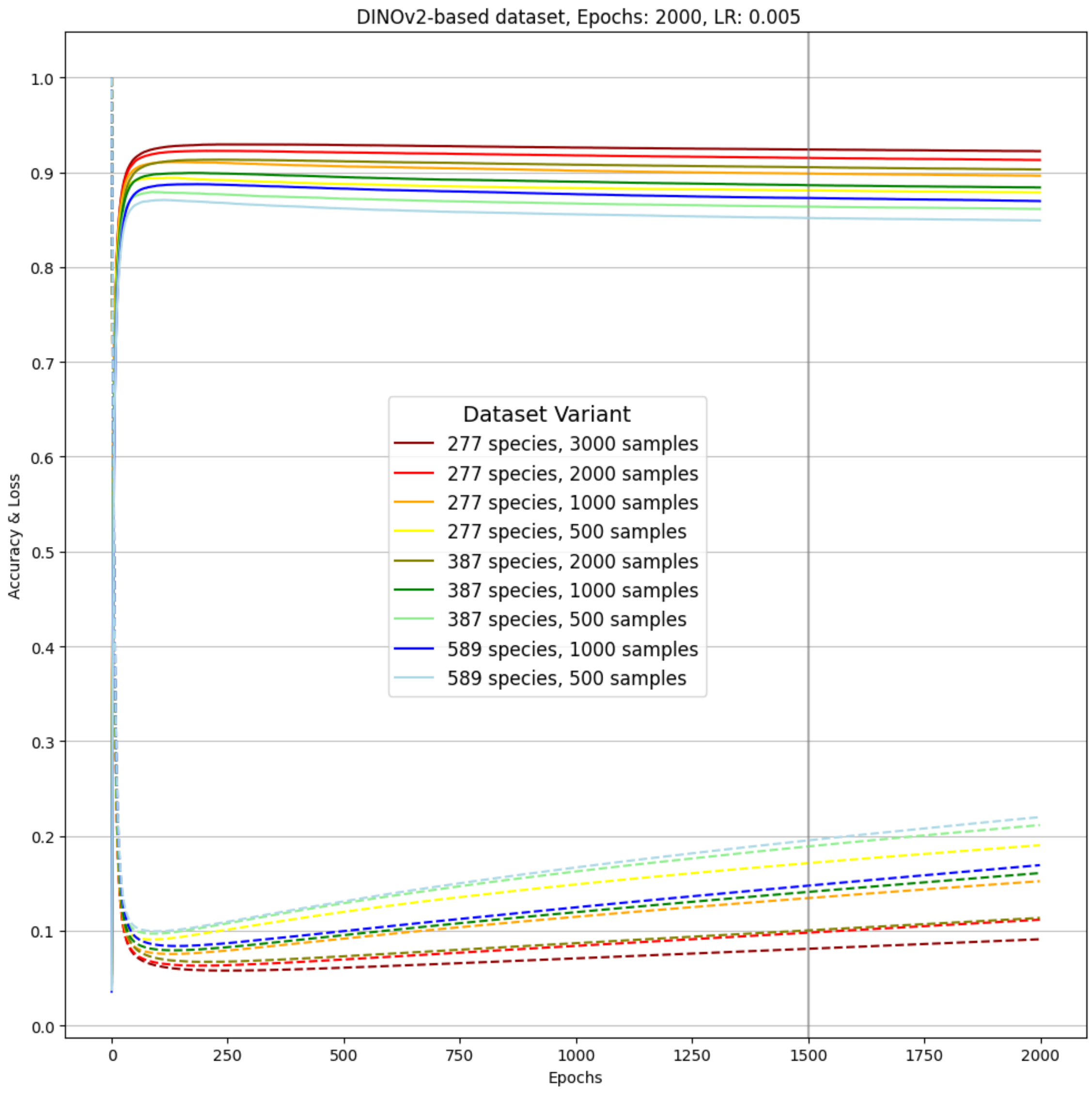


Figure 27: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine DINOv2-based dataset variants with 2000 epochs and a learning rate of 0.005.

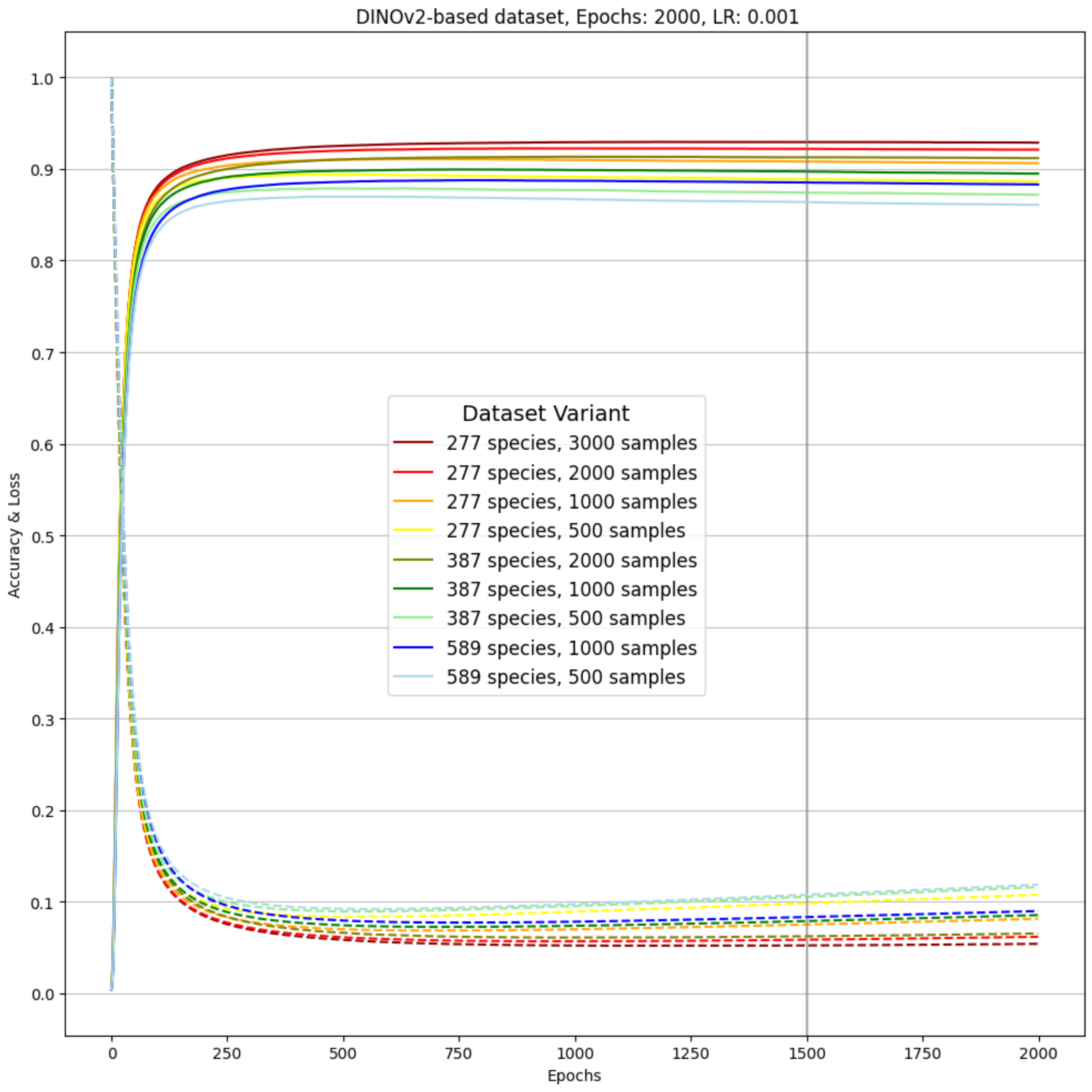


Figure 28: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine DINOv2-based dataset variants with 2000 epochs and a learning rate of 0.001.

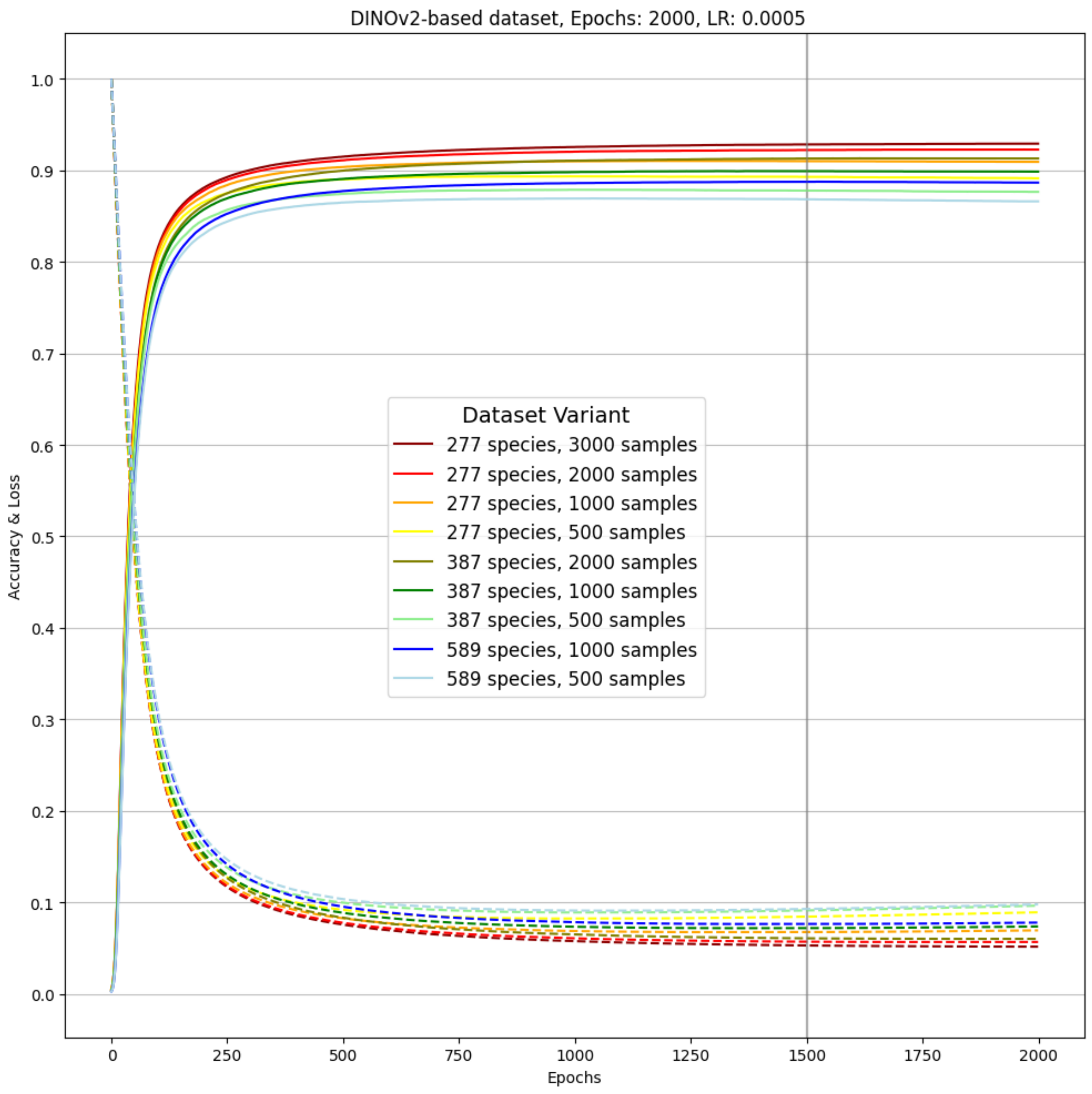


Figure 29: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine DINOv2-based dataset variants with 2000 epochs and a learning rate of 0.0005.

Cross-validation results for Logistic Regression, averaged over ResNet-50-based dataset variants

Choice	**Learning Rate**	**Epochs**	**Mean Accuracy (%) ± Std (%)**
-	0.0010	2000	**64.90** ± 4.42
x	0.0010	1500	64.82 ± 4.16
-	0.0005	2000	64.43 ± 3.91
-	0.0050	1500	63.96 ± 5.39
-	0.0005	1500	63.76 ± **3.71**
-	0.0050	2000	63.61 ± 5.65

Table 14: The table is sorted by mean accuracy, the highest mean accuracy and lowest standard deviation are marked in bold.

Cross-validation results for KNN, averaged over ResNet-50-based dataset variants

Choice	**Neighbors**	**Mean Accuracy (%) ± Std (%)**
-	35	**38.31** ± 5.14
x	50	38.19 ± 5.03
-	65	37.96 ± **4.98**

Table 15: The table is sorted by mean accuracy, the highest mean accuracy and lowest standard deviation are marked in bold.

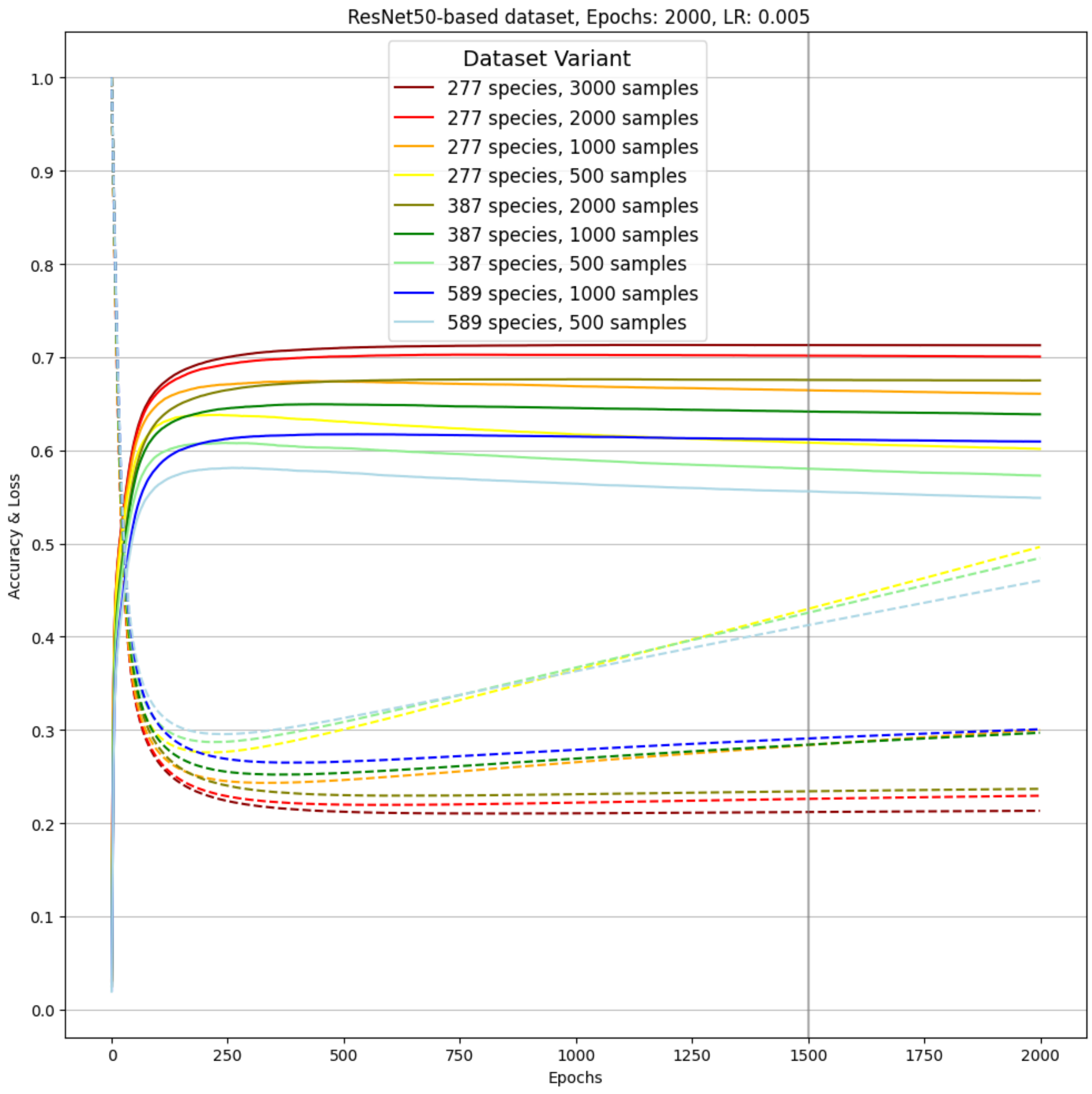


Figure 30: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine ResNet-50-based dataset variants with 2000 epochs and a learning rate of 0.005.

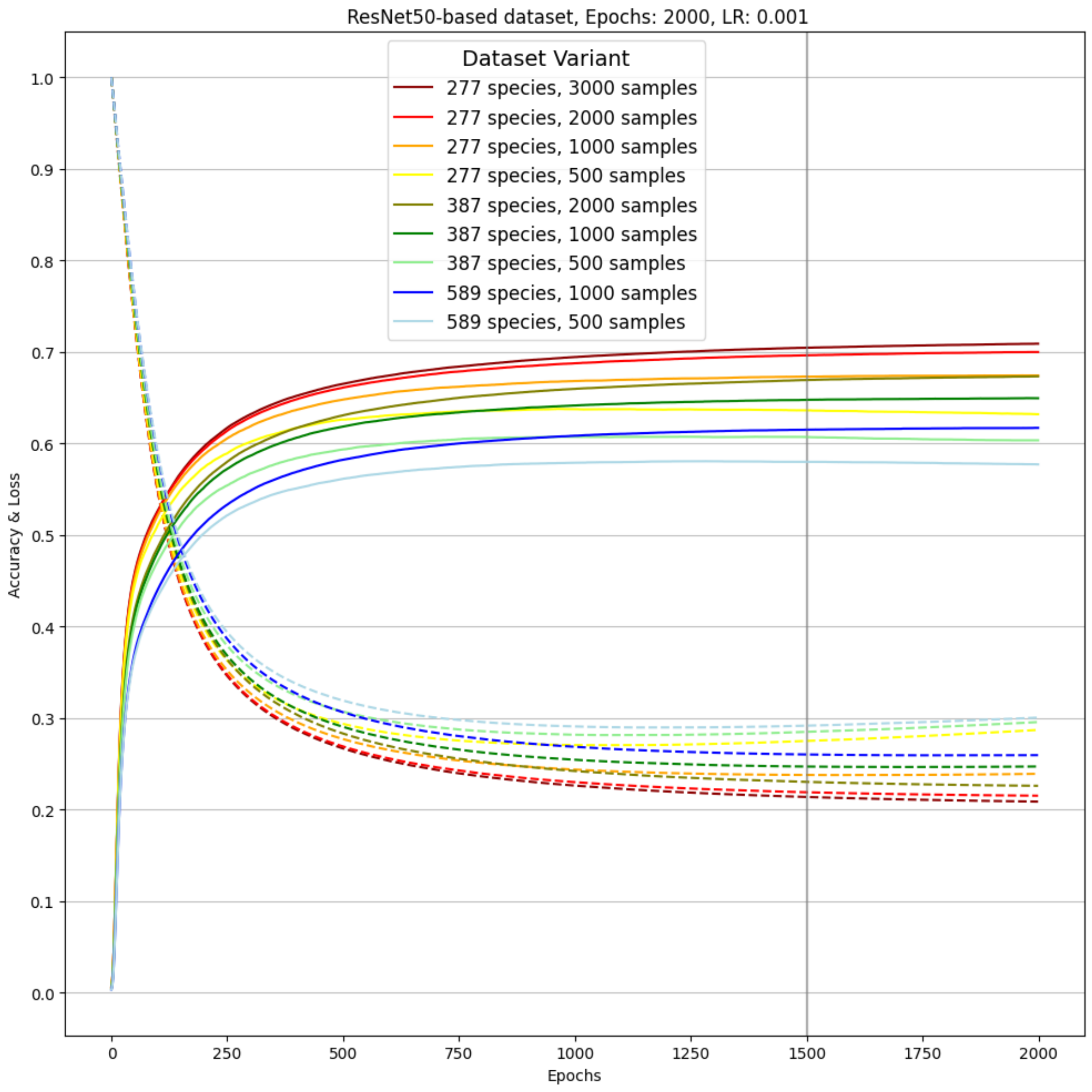


Figure 31: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine ResNet-50-based dataset variants with 2000 epochs and a learning rate of 0.001.

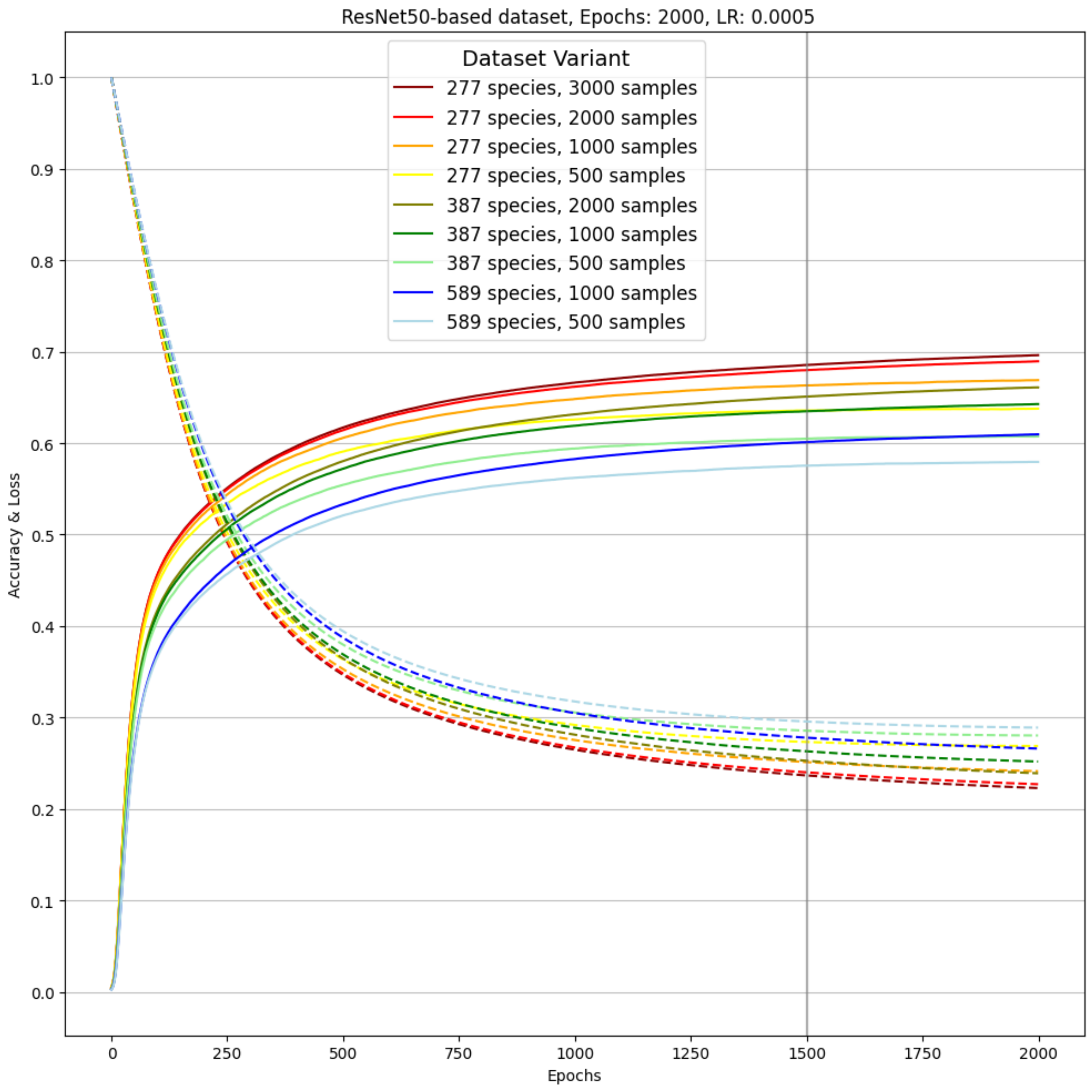


Figure 32: Validation loss (dashed lines) and validation accuracies (continuous lines) for Logistic Regression Classifier for the nine ResNet-50-based dataset variants with 2000 epochs and a learning rate of 0.0005.

B Appendix: Cross-Validation Results

Species Amount	Sample Amount	DINOv2 LR	DINOv2 KNN	ResNet-50 LR	ResNet-50 KNN
277	**3,000**	92.78	75.01	70.34	45.88
	2,000	92.27	73.42	69.58	43.99
	1,000	90.94	70.43	67.61	40.67
	500	89.19	66.94	63.68	36.96
387	**2,000**	91.26	71.10	66.79	40.75
	1,000	89.87	67.95	64.65	37.38
	500	87.99	64.15	61.22	33.81
589	**1,000**	88.93	65.46	61.62	33.87
	500	86.77	61.50	57.81	30.49

Table 16: Mean accuracies averaged over 5 cross-validation splits in %, for all nine dataset variants and four FM-CLF combinations.

Min, Mean and Max were applied on all mean scores (averaged over results of the 5 cross-validation splits) of all 36 cross-validation runs (involving all dataset variants and FM-CLF combinations.)

Metric	Min Gap (%)	Mean Gap (%)	Max Gap (%)
Precision	-0.10	0.85	2.62
Recall	0.00	0.00	0.00
F1-Score	-0.71	-0.17	0.08

Table 17: Average differences (*gaps*) between Accuracy and Precision, Recall and F1-Score. A negative score indicates, that the metric was smaller than the accuracy.

Calculated over corresponding 5 validation folds over all combinations of classifiers and dataset variants

Metric	Min Std (%)	Mean Std (%)	Max Std (%)
Accuracy	0.04	0.15	0.32
Precision	0.07	0.16	0.32
Recall	0.04	0.15	0.32
F1-Score	0.06	0.15	0.32

Table 18: Standard deviation statistics for Accuracy, Precision, Recall, and F1-Score in between 5 validation folds.

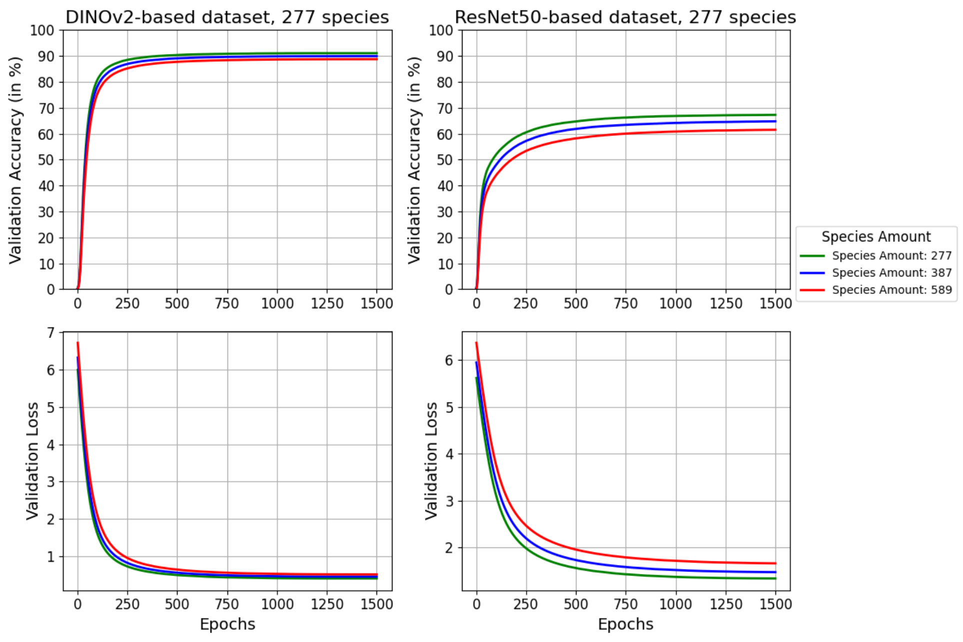


Figure 33: Difference in validation accuracies and losses for dataset variants with varying amount of species while the amount of samples stays constant at 1,000. The curves represent results averaged across all 5 cross-validation splits.

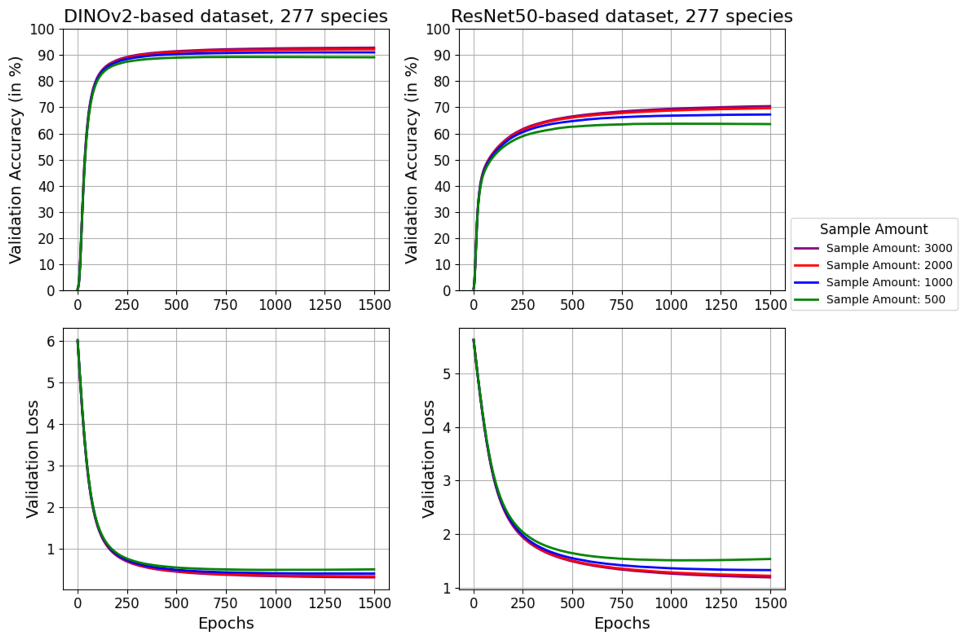


Figure 34: Difference in validation accuracies and losses for datasets with varying amount of samples while amount of species stays constant at 277. The curves represent results averaged across all 5 cross-validation splits.

C Appendix: Confidence Analysis

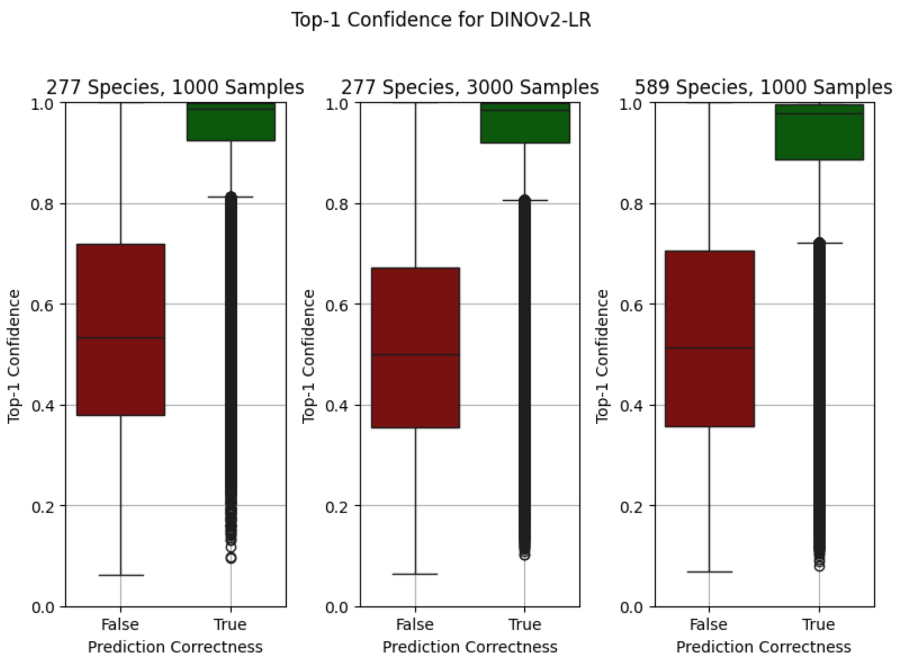


Figure 35: Confidences for correct and incorrect predictions on the three key dataset for the DINOv2-LR combination.

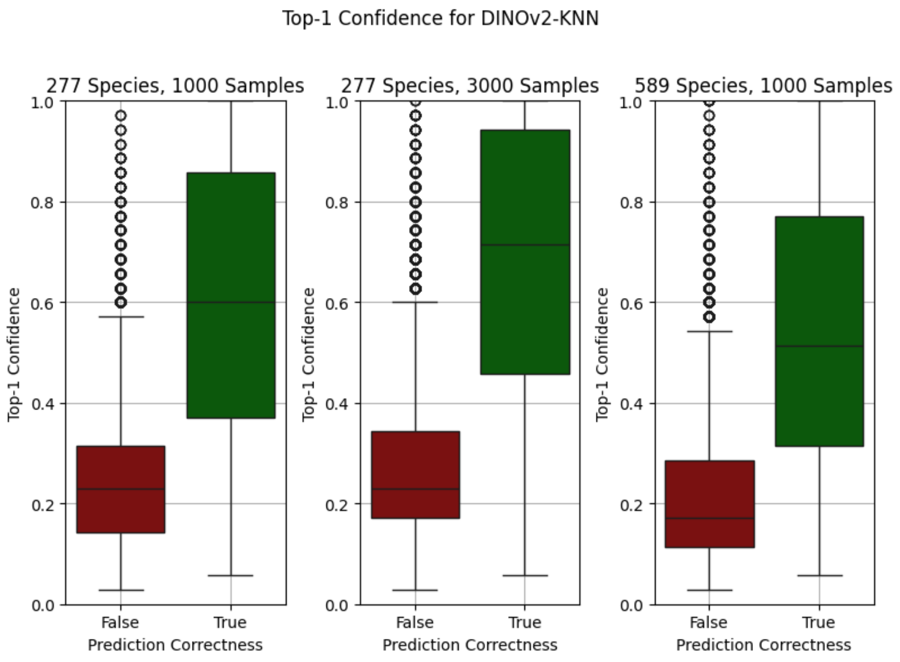


Figure 36: Confidences for correct and incorrect predictions on the three key dataset for the DINOv2-KNN combination.

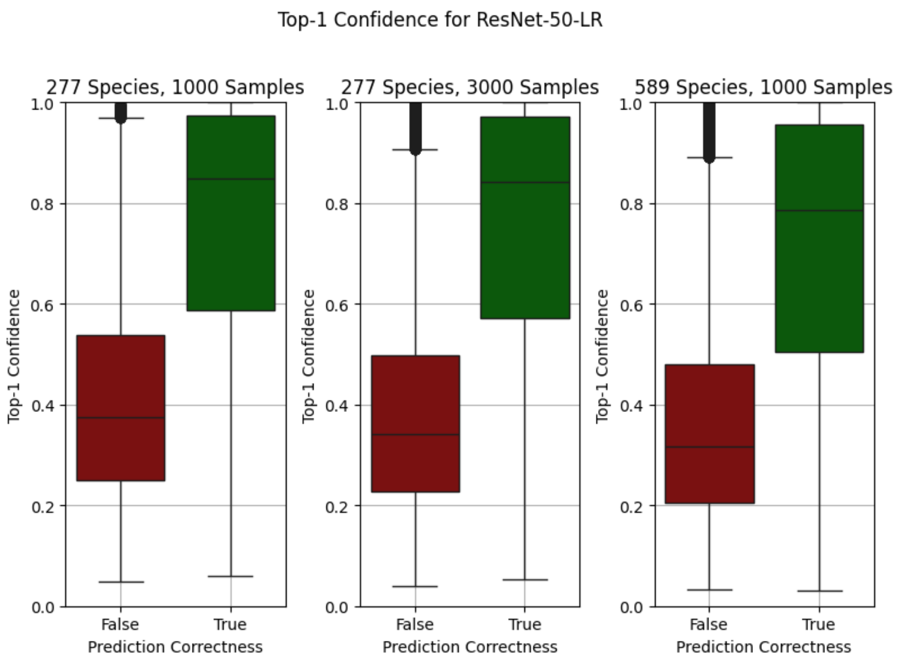


Figure 37: Confidences for correct and incorrect predictions on the three key dataset for the ResNet-50-LR combination.

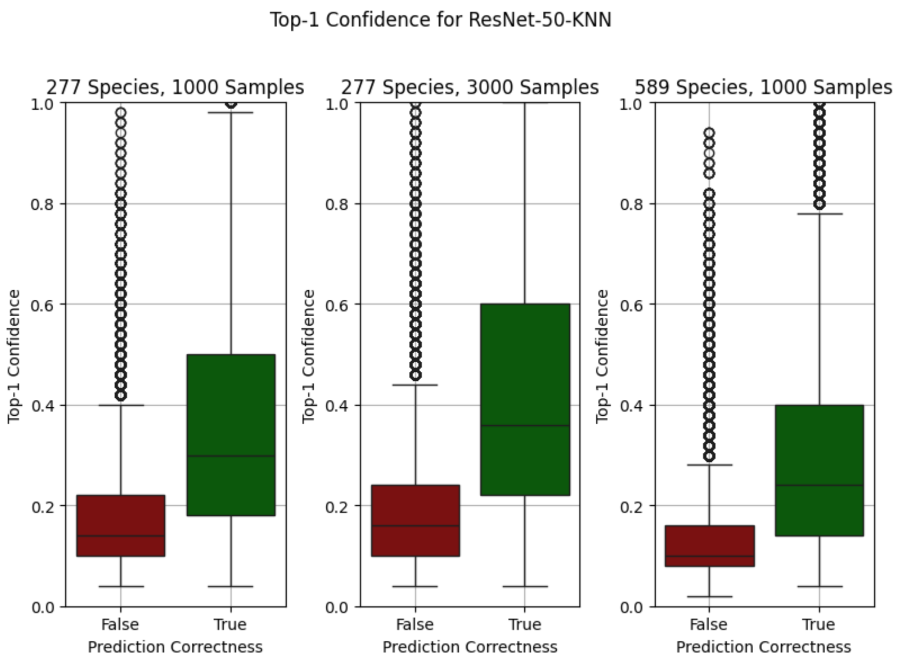


Figure 38: Confidences for correct and incorrect predictions on the three key dataset for the ResNet-50-KNN combination.

D Appendix: Species-Level Analysis

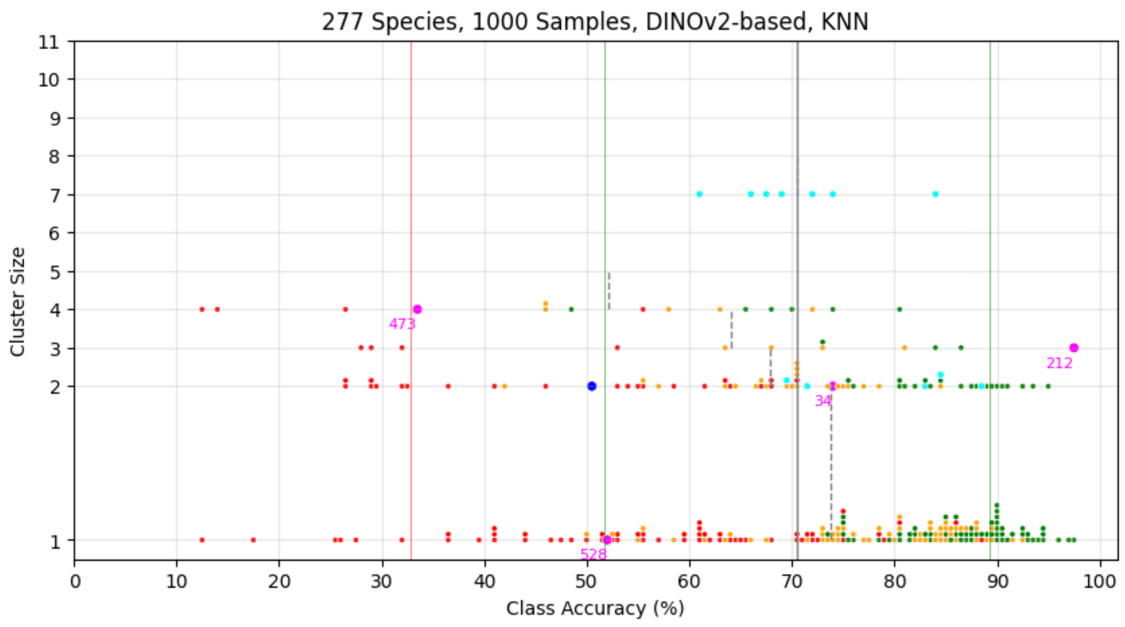


Figure 39: Class accuracy distribution for the reference dataset using the DINOv2-KNN combination.

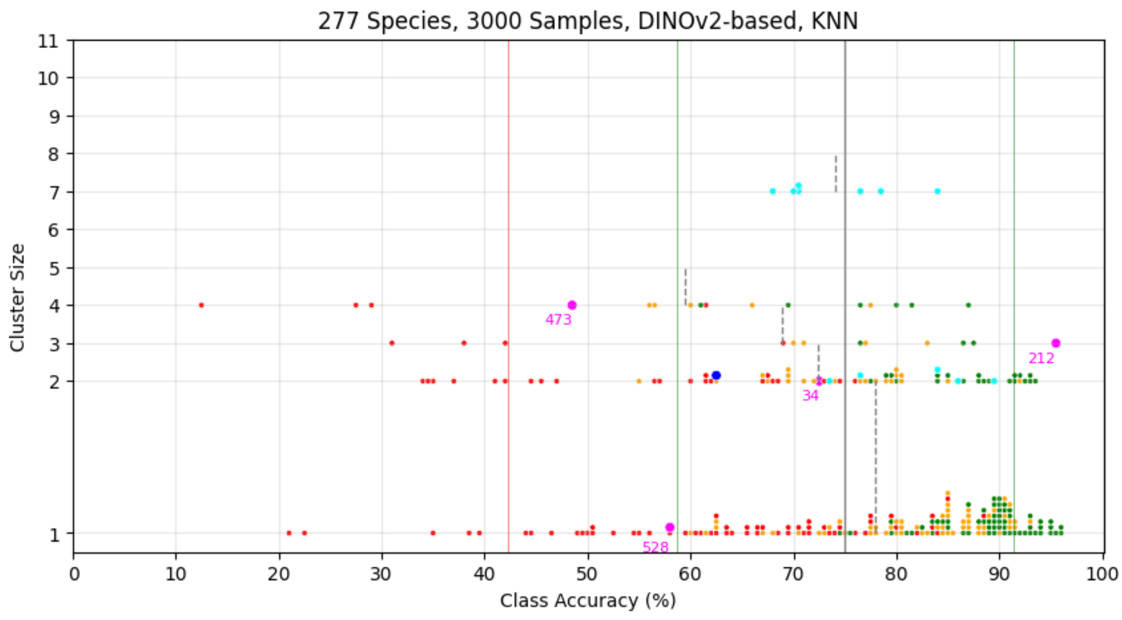


Figure 40: Class accuracy distribution for the sample dataset using the DINOv2-KNN combination.

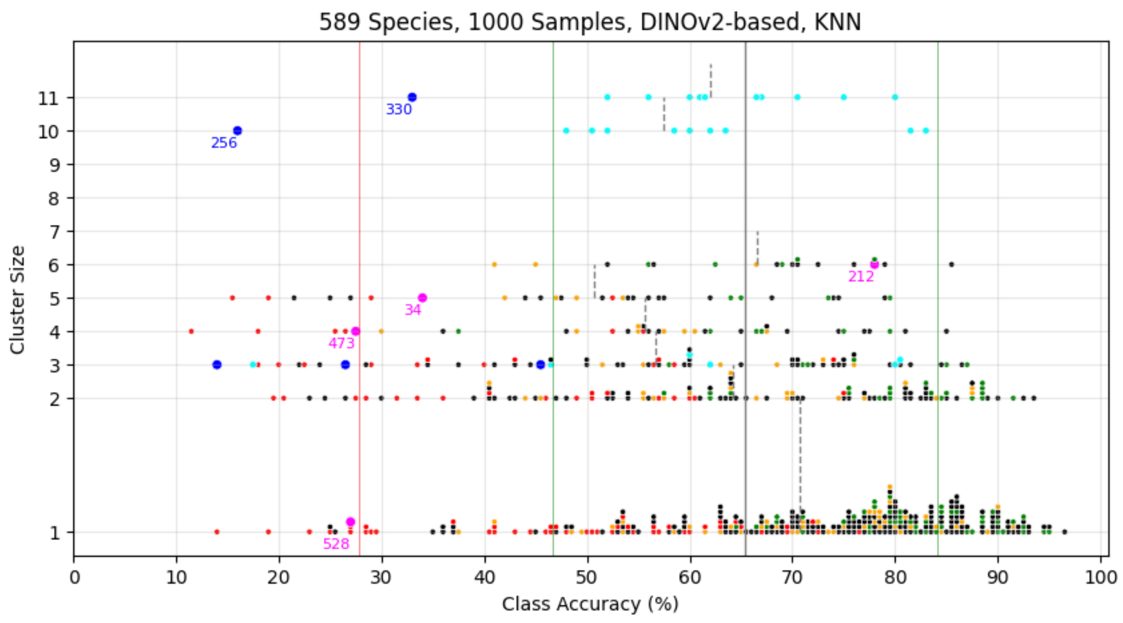


Figure 41: Class accuracy distribution for the species dataset using the DINOv2-KNN combination. New species in this dataset are marked as black dots.

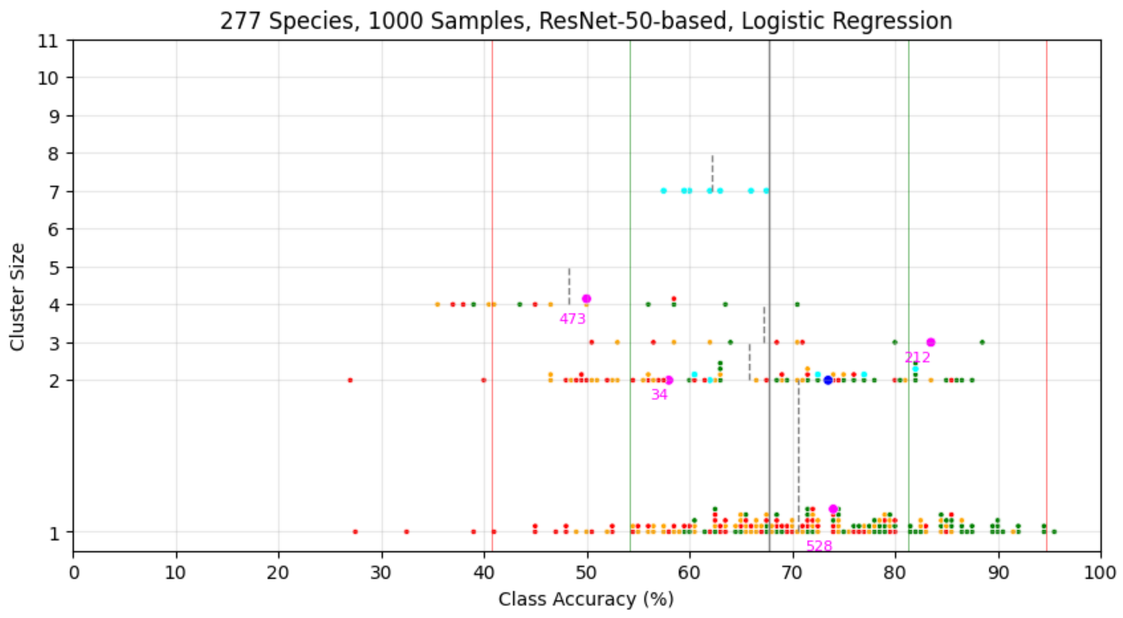


Figure 42: Class accuracy distribution for the reference dataset using the ResNet-50-LR combination.

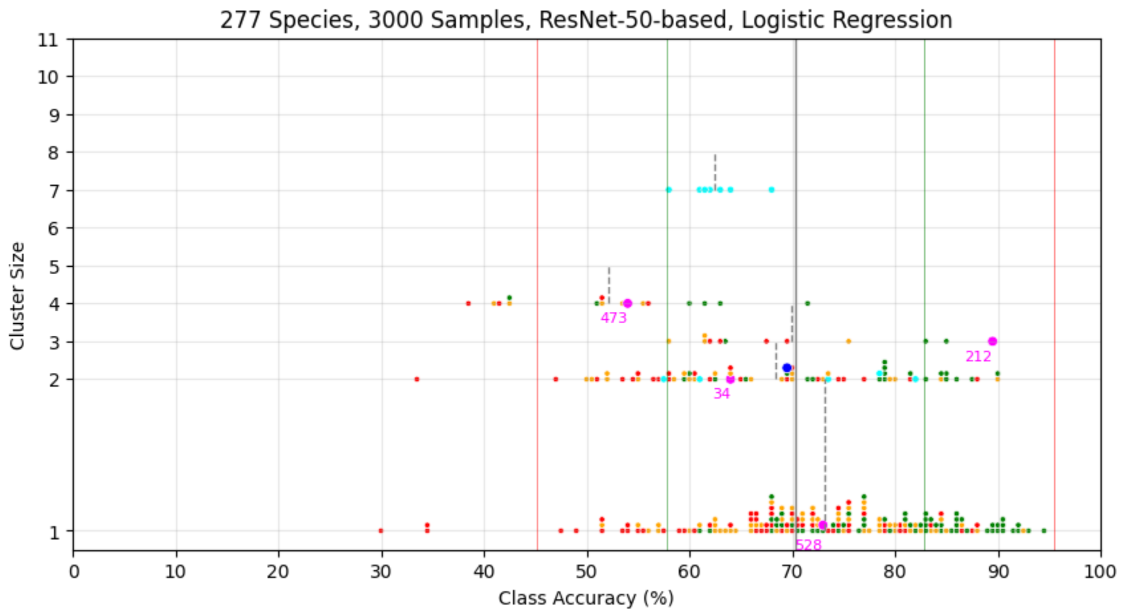


Figure 43: Class accuracy distribution for the sample dataset using the ResNet-50-LR combination.

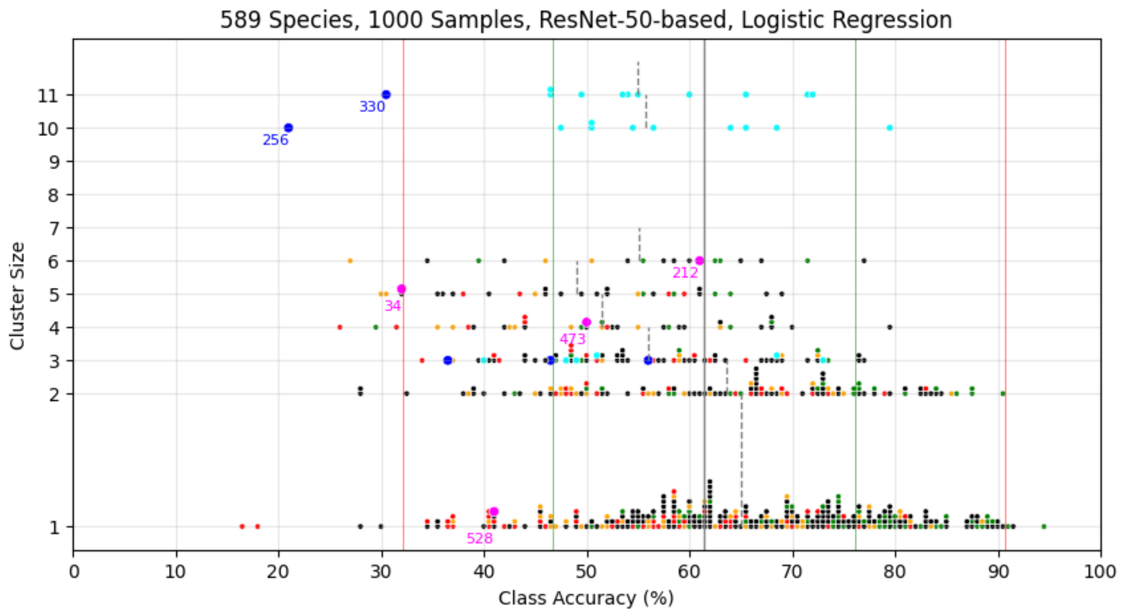


Figure 44: Class accuracy distribution for the species dataset using the ResNet-50-LR combination. New species in this dataset are marked as black dots.

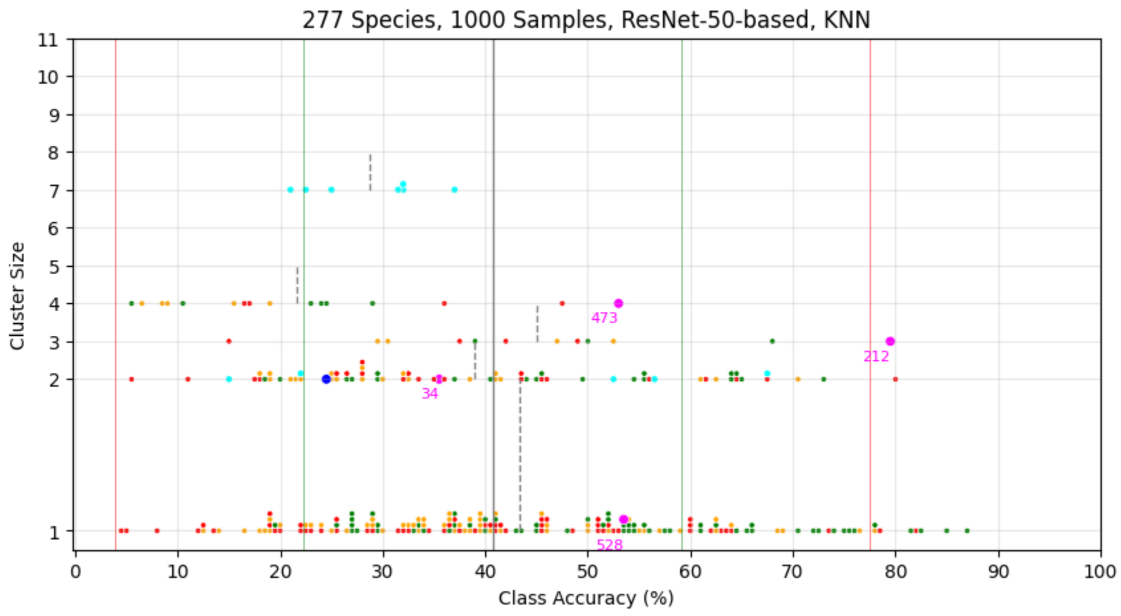


Figure 45: Class accuracy distribution for the reference dataset using the ResNet-50-KNN combination.

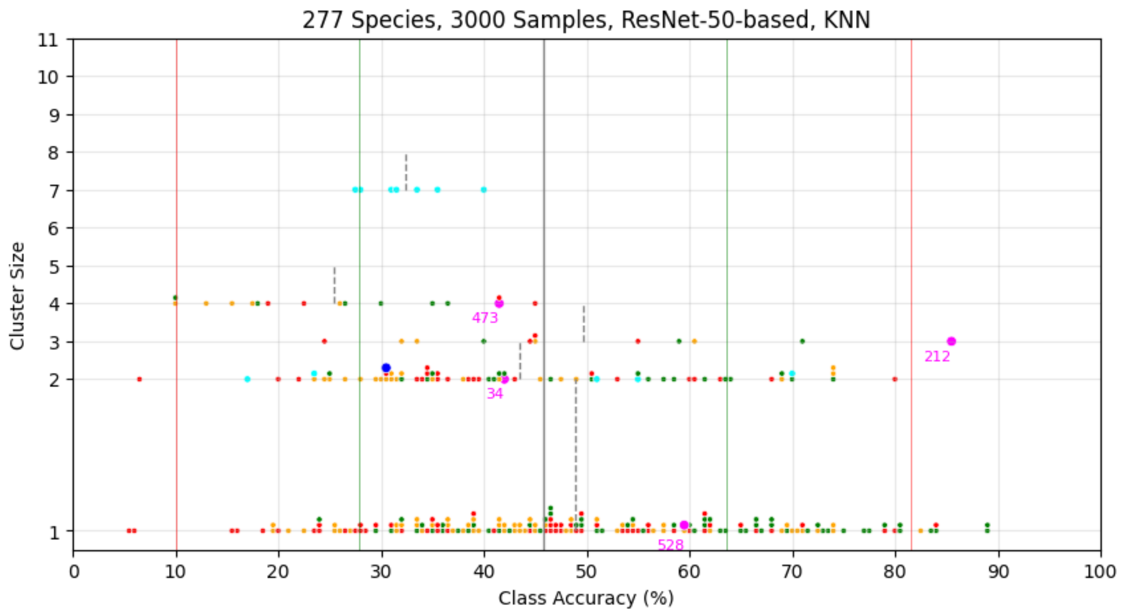


Figure 46: Class accuracy distribution for the sample dataset using the ResNet-50-KNN combination.

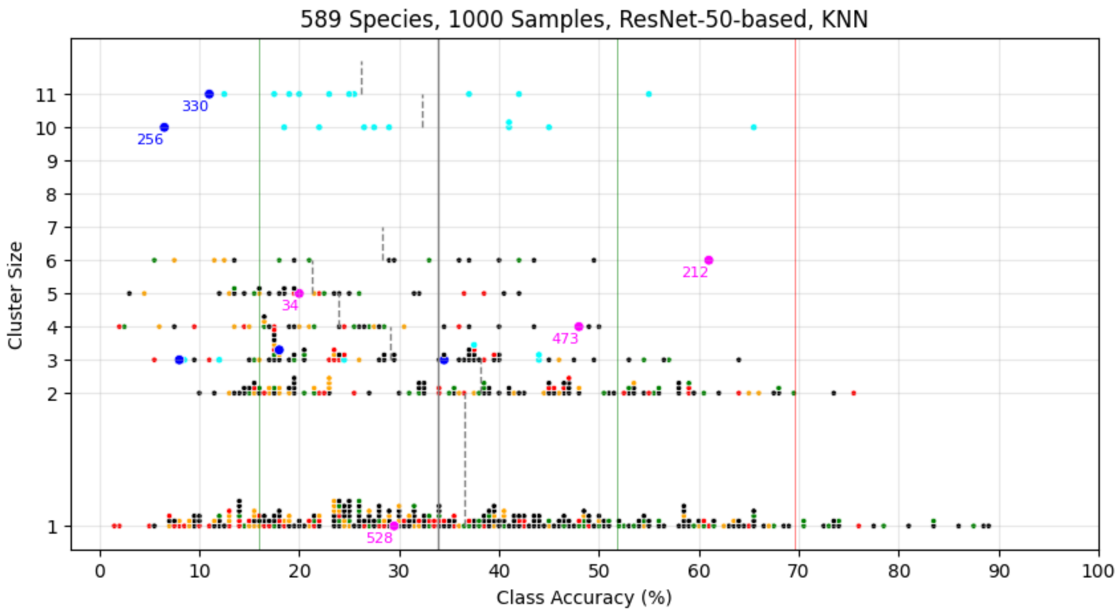


Figure 47: Class accuracy distribution for the species dataset using the ResNet-50-KNN combination. New species in this dataset are marked as black dots.

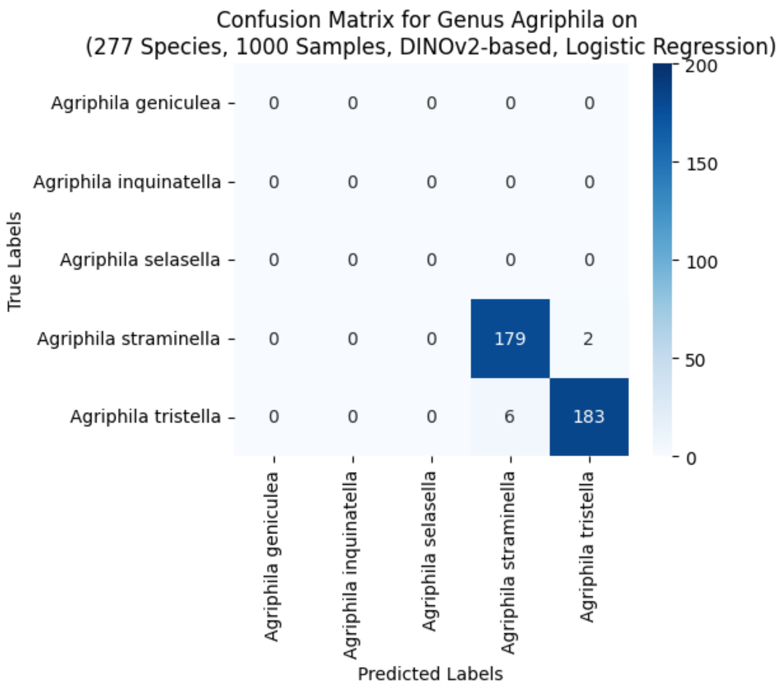


Figure 48: Confusion Matrix of Agriphila Genus on the reference dataset using the DINOv2-LR combination.

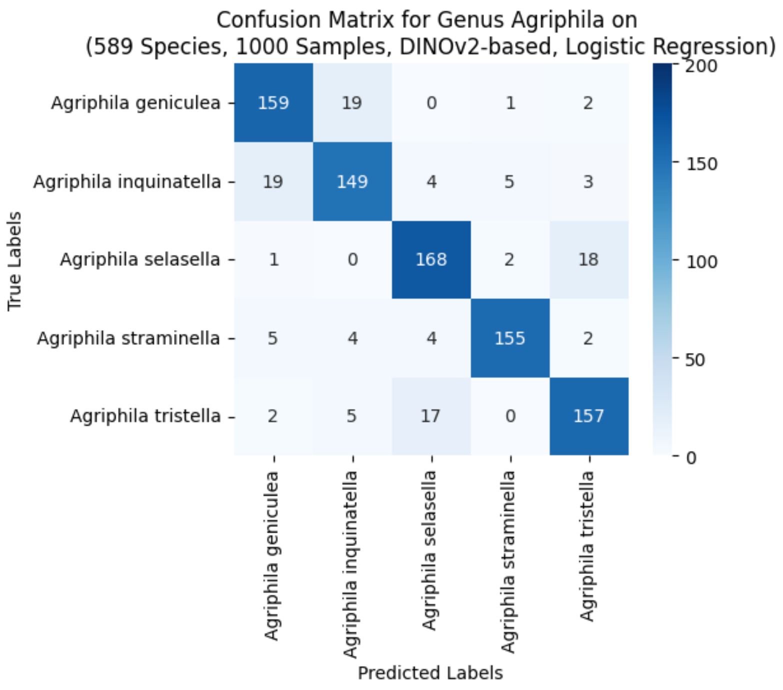


Figure 49: Confusion Matrix of Agriphila Genus on the species dataset using the DINOv2-LR combination.

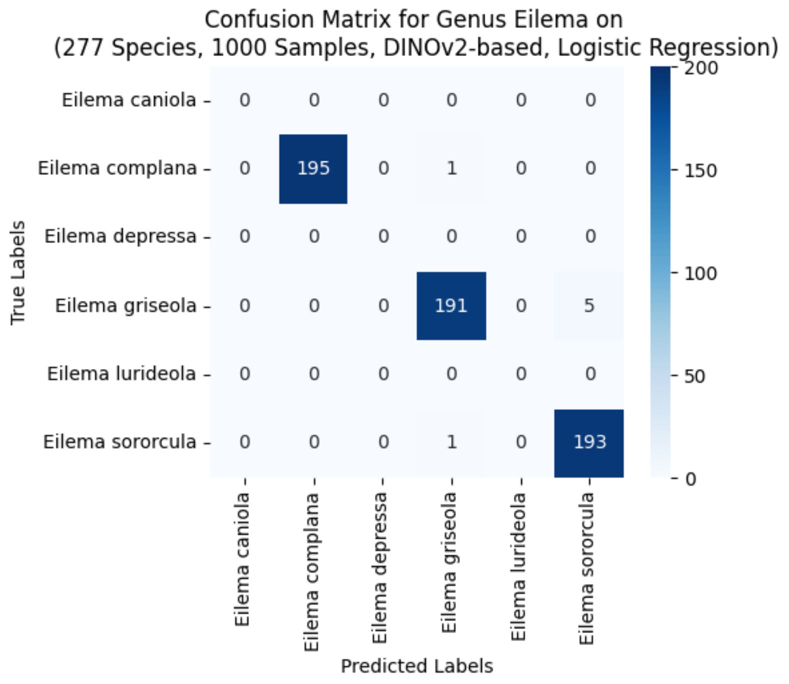


Figure 50: Confusion Matrix of Eilema Genus on the reference dataset using the DINOv2-LR combination.

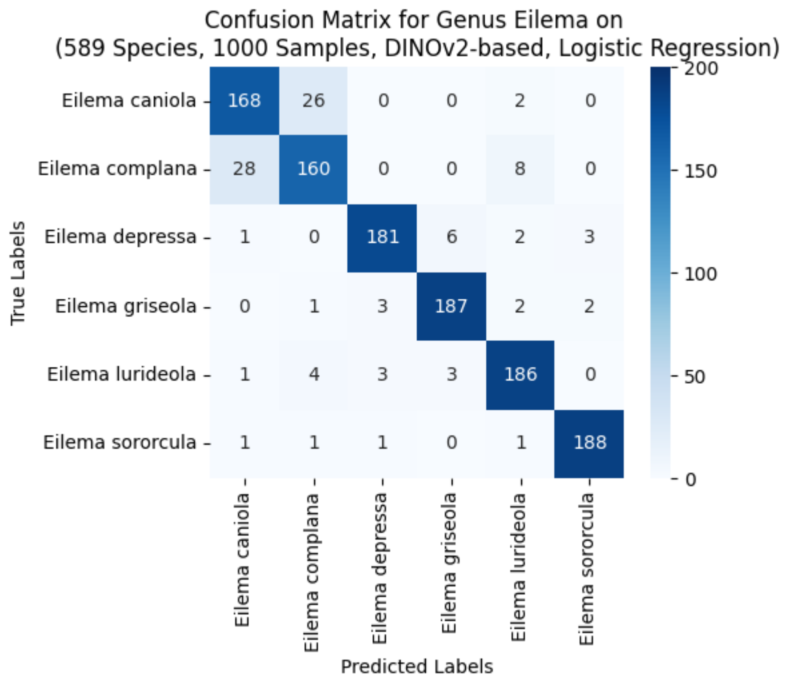


Figure 51: Confusion Matrix of Eilema Genus on the species dataset using the DINOv2-LR combination.

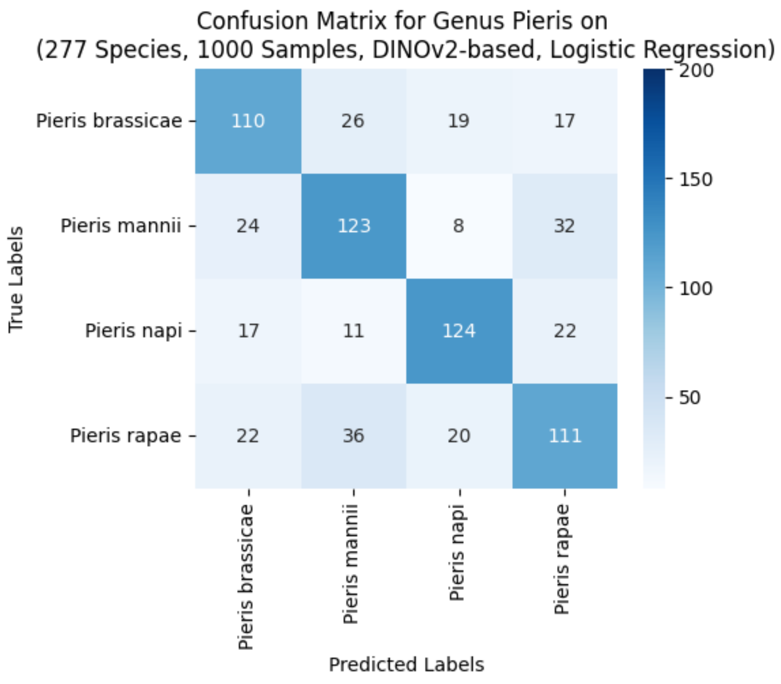


Figure 52: Confusion Matrix of the Pieris Genus on the reference dataset using the DINOv2-LR combination.

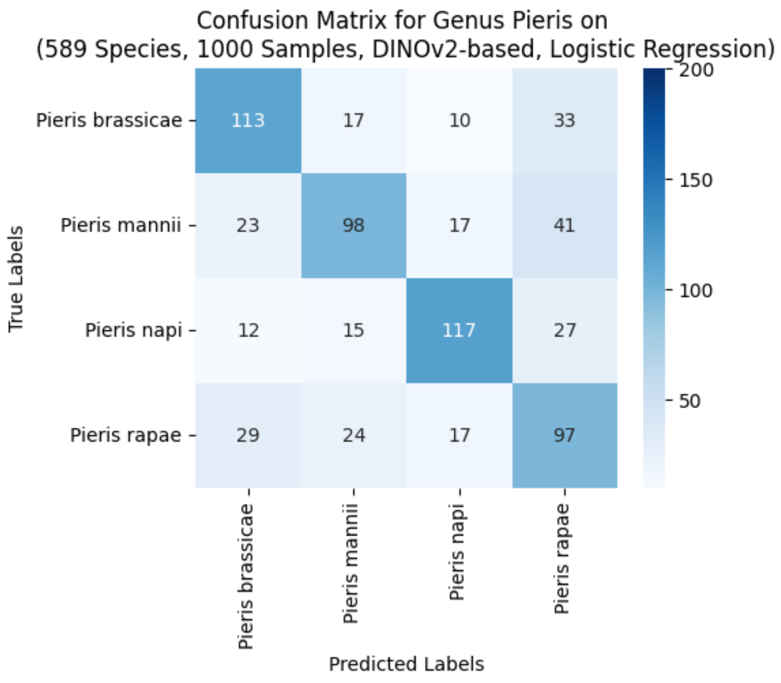


Figure 53: Confusion Matrix of the Pieris Genus on the species dataset using the DINOv2-LR combination.

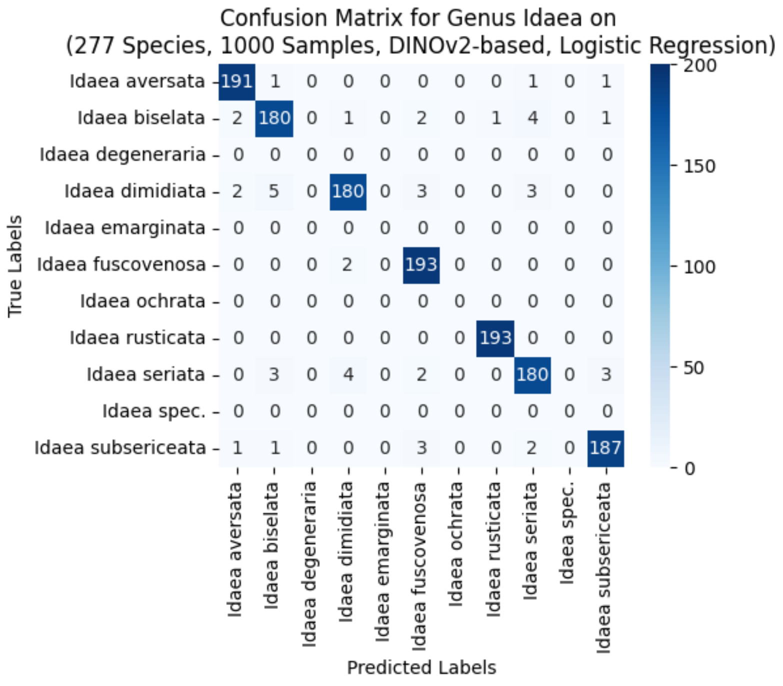


Figure 54: Confusion Matrix of the Idaea Genus on the reference dataset using the DINOv2-LR combination.

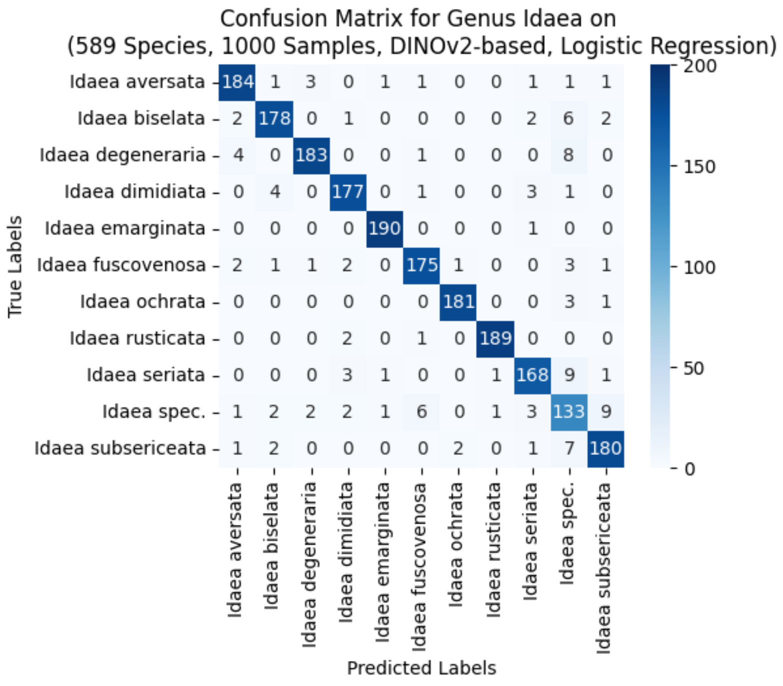


Figure 55: Confusion Matrix of the Idaea Genus on the species dataset using the DINOv2-LR combination.

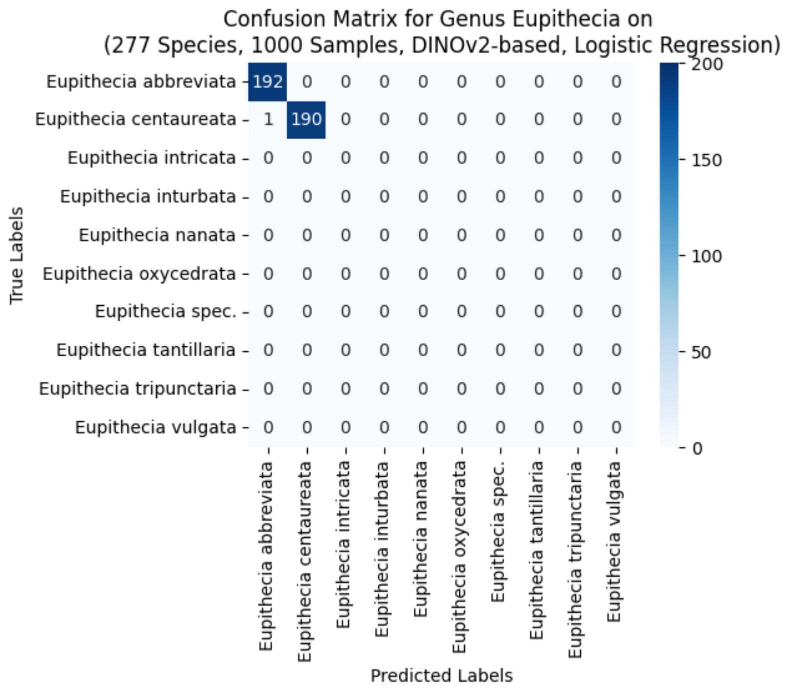


Figure 56: Confusion Matrix of the Eupithecia Genus on the reference dataset using the DINOv2-LR combination.

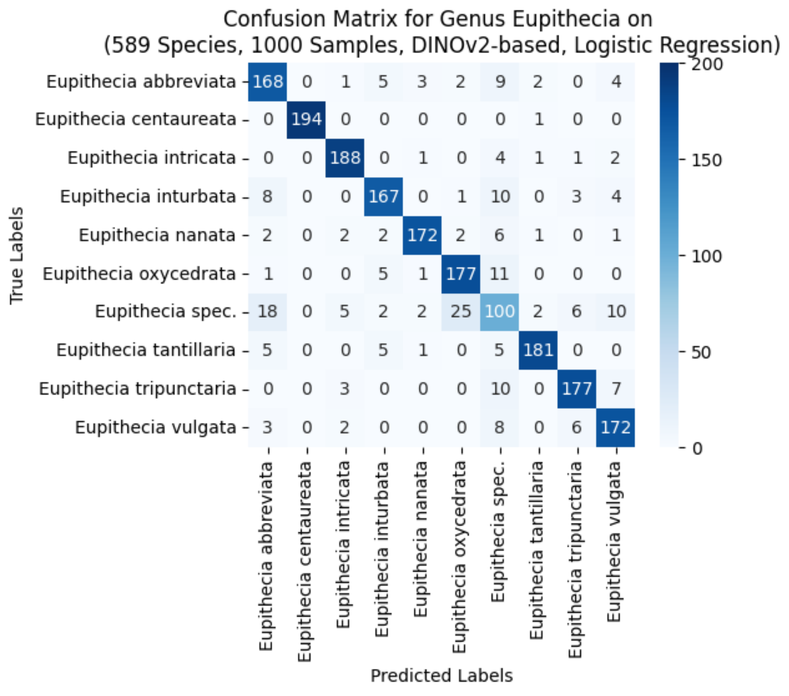


Figure 57: Confusion Matrix of the Eupithecia Genus on the species dataset using the DINOv2-LR combination.

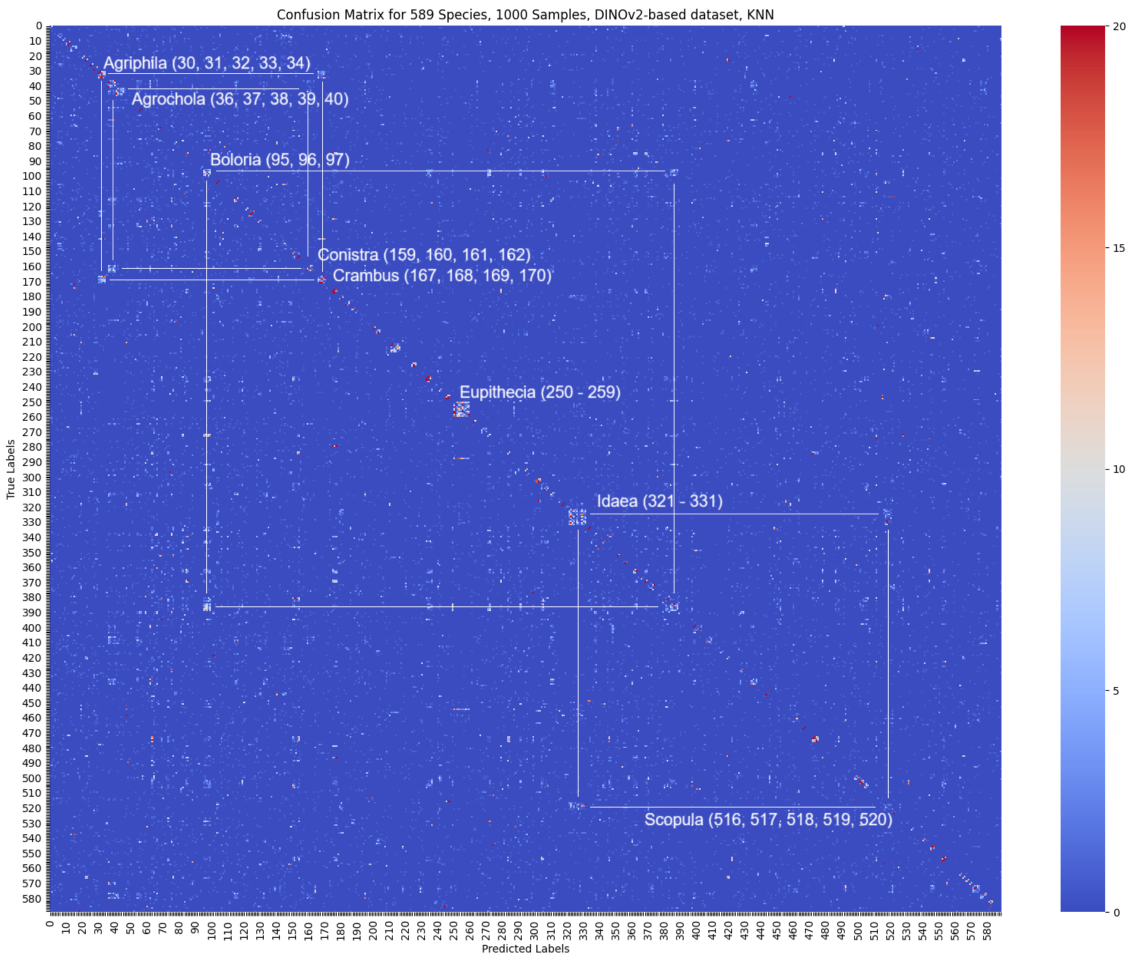


Figure 58: Confusion matrix including all 589 species (sorted after name and therefore grouped by genus) of the species dataset for KNN trained on DINOv2 embeddings (DINOv2-KNN). With correct classification numbers being set to zero, the focus lies on the misclassifications. The corners of the squares indicate which genus clusters are confused with each other.

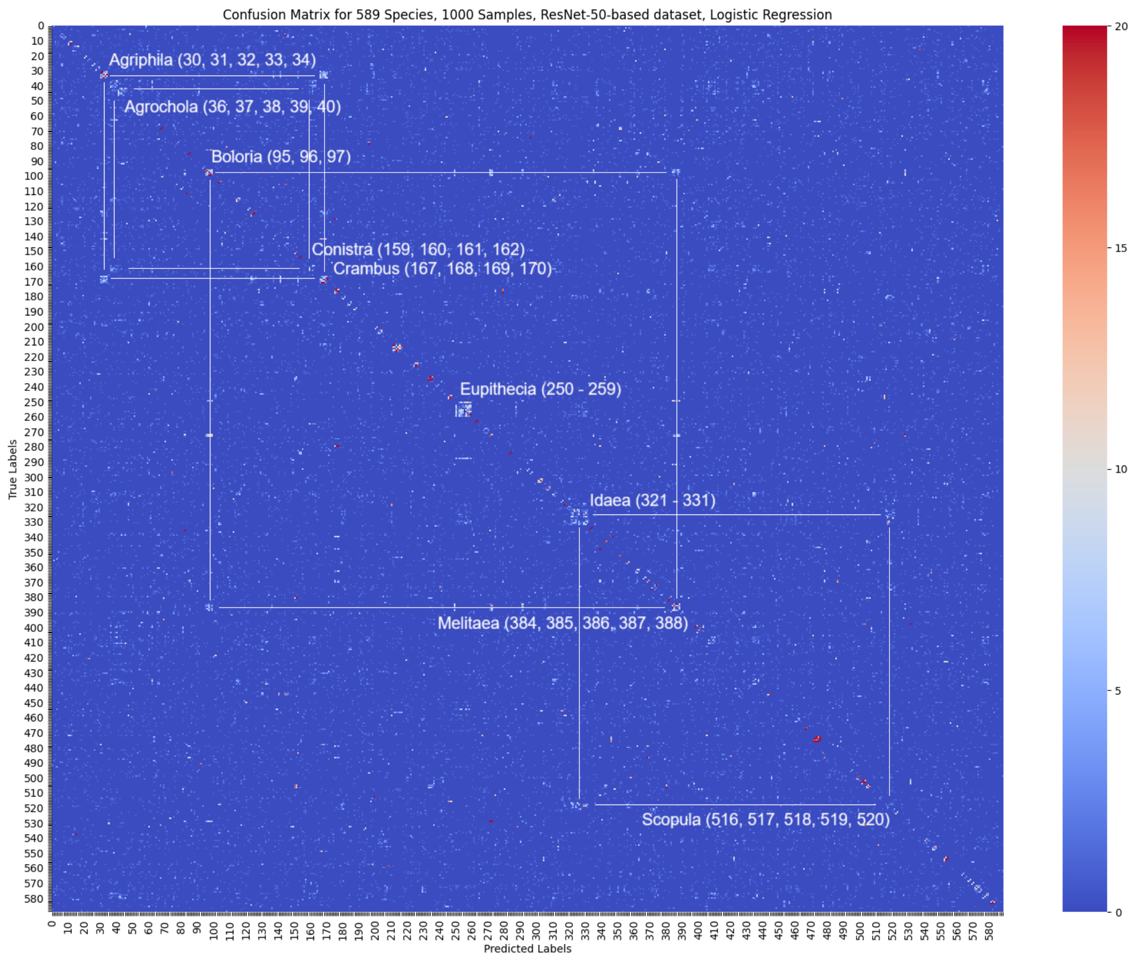


Figure 59: Confusion matrix including all 589 species (sorted after name and therefore grouped by genus) of the species dataset for Logistic Regression trained on ResNet-50 embeddings (ResNet-50-LR). With correct classification numbers being set to zero, the focus lies on the misclassifications. The corners of the squares indicate which genus clusters are confused with each other.

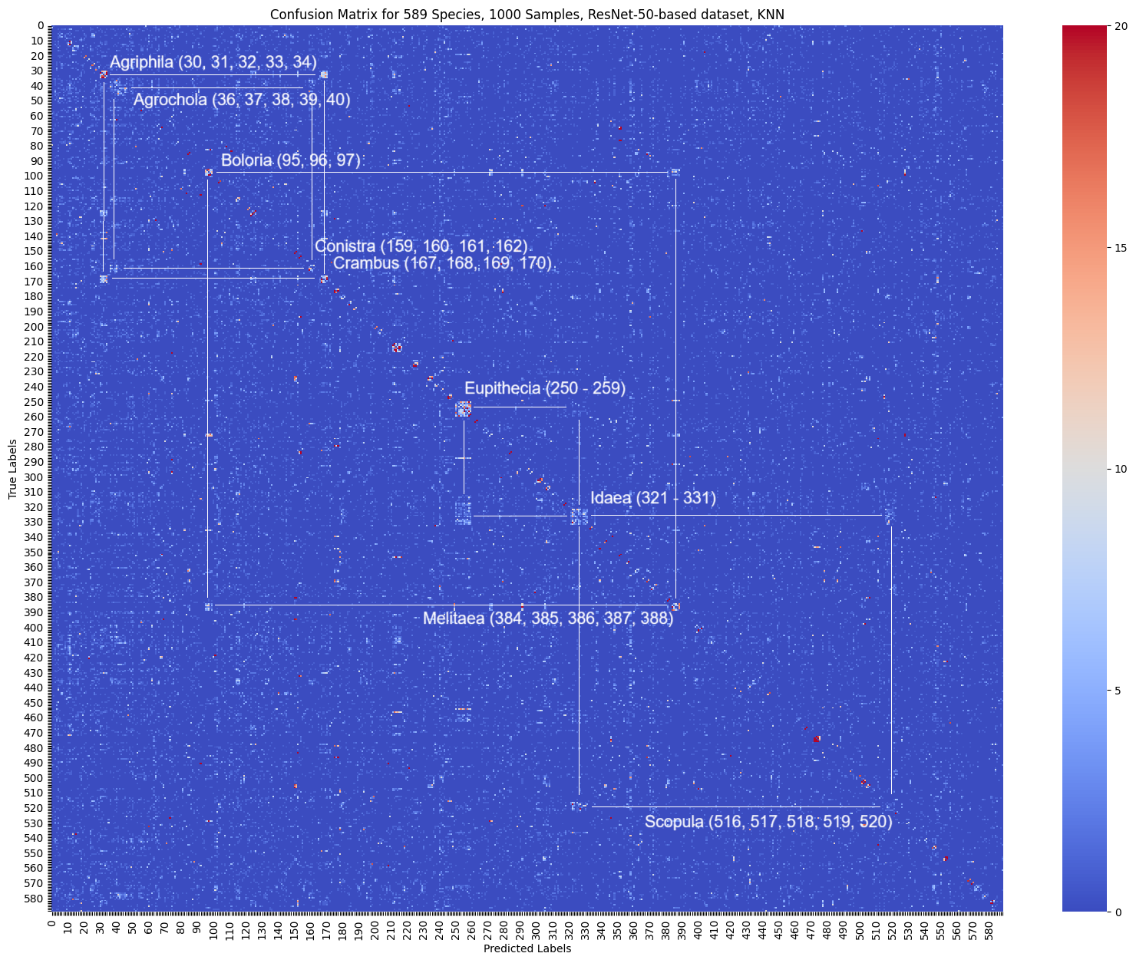


Figure 60: Confusion matrix including all 589 species (sorted after name and therefore grouped by genus) of the species dataset for KNN trained on ResNet-50 embeddings (ResNet-50-KNN). With correct classification numbers being set to zero, the focus lies on the misclassifications. The corners of the squares indicate which genus clusters are confused with each other.

Bibliography

Sample: Species - Crambidae indet., a. URL https://observation.org/photos/30399870.jpg. gbifID: 4890526603, Accessed: 2025-02-25.

Sample: Species - Eilema caniola, b. URL https://observation.org/photos/69618055.jpg. gbifID: 4160499365, Accessed: 2025-02-25.

Sample: Species - Eilema complana, c. URL https://observation.org/photos/54551194.jp. gbifID: 3997271114, Accessed: 2025-02-25.

Sample: Species - Eilema lurideola, d. URL https://observation.org/photos/22010983.jpg. gbifID: 2837482917, Accessed: 2025-02-25.

Sample: Species - Fabriciana adippe, e. URL https://observation.org/photos/55899069.jpg. gbifID: 3997395183, Accessed: 2025-02-25.

Sample: Species - Fabriciana niobe, f. URL https://observation.org/photos/52058002.jpg. gbifID: 3904764374, Accessed: 2025-02-25.

Sample: Species - Argiphila selasella, g. URL https://observation.org/photos/86728263.jpg. gbifID: 4881198101, Accessed: 2025-02-25.

Sample: Species - Agriphila tristella, h. URL https://observation.org/photos/54812633.jpg. gbifID: 3717772501, Accessed: 2025-02-25.

Sample: Species - Pieris brassicae, i. URL https://observation.org/photos/75713564.jpg. gbifID: 4409396785, Accessed: 2025-02-25.

Sample: Species - Pieris mannii, j. URL https://observation.org/photos/88743427.jpg. gbifID: 4883560699, Accessed: 2025-02-25.

Sample: Species - Pieris rapae, k. URL https://observation.org/photos/71777109.jpg. gbifID: 4409299347, Accessed: 2025-02-25.

Sample: Species - Speyeria aglaja, l. URL https://observation.org/photos/71802906.jpg. gbifID: 4409388452, Accessed: 2025-02-25.

Sample: Species - Eupithecia spec., m. URL https://observation.org/photos/86654223.jpg. gbifID: 4883241740, Accessed: 2025-02-25.

Sample: Species - Eupithecia oxycedrata, n. URL https://observation.org/photos/87361092.jpg. gbifID: 4881370136, Accessed: 2025-02-25.

Sample: Species - Eupithecia vulgata, o. URL https://observation.org/photos/8716009.jpg. gbifID: 1976898040, Accessed: 2025-02-25.

Sample: Moth out of focus, p. URL https://observation.org/photos/30566083.jpg. gbifID: 3726205023, Accessed: 2025-02-25.

Sample: Butterfly behind grass, q. URL `https://observation.org/photos/27621411.jpg`. gbifID: 3725704126, Accessed: 2025-02-25.

Sample: Multiple Insects, r. URL `https://observation.org/photos/32456182.jpg`. gbifID: 3728033111, Accessed: 2025-02-25.

Sample: Small moth, s. URL `https://observation.org/photos/58375000.jpg`. gbifID: 4056593551, Accessed: 2025-02-25.

Sample: Moth Trap, t. URL `https://observation.org/photos/17304777.jpg`. gbifID: 1966132886, Accessed: 2025-02-25.

GBIF, 2025. URL `https://gbif.org/`. Accessed: 2025-01-29.

Observation.org, 2025. URL `https://observation.org/`. Accessed: 2025-01-29.

Daniel Berrar et al. Cross-validation., 2019.

Qi Chang, Hui Qu, Pengxiang Wu, and Jingru Yi. Fine-grained butterfly and moth classification using deep convolutional neural networks. *Rutgers University: New Brunswick, NJ, USA*, 2017.

Cordis.Europa.Eu Cordis. Boosting biodiversity research with citizen science, 4 2023. URL `https://cordis.europa.eu/article/id/443300-boosting-biodiversity-research-with-citizen-science`.

Encord. DINOv2: Self-supervised Learning Model Explained, 2024. URL `https://encord.com/blog/dinov2-self-supervised-learning-explained/`. Accessed: 2025-02-20.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International conference on learning representations*, 2018.

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Uniform manifold approximation and projection (umap) and its variants: tutorial and survey. *arXiv preprint arXiv:2109.02508*, 2021.

G Hajizadeh, H Jalilvand, MR Kavosi, and HB Varandi. Relationship between vegetation characteristics and lepidoptera diversity in the hyrcanian forest, iran (insecta: Lepidoptera). *SHILAP Revista de lepidopterología*, 50(199):561–574, 2022.

K. Hartmann, J. Krois, and A. Rudolph. *Statistics and Geodata Analysis using R (SOGA-R)*. Department of Earth Sciences, Freie Universitaet Berlin, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

John B Heppner. *Manual of Lepidoptera: Classification of the Butterflies and Moths of the World*. CRC, 2010.

IBM. What is the k-nearest neighbors (KNN) algorithm?, 2025. URL `https://www.ibm.com/think/topics/knn`. Accessed: 2025-02-10.

IBM. Knn diagram, 2025. URL `https://www.ibm.com/content/dam/connectedassets-adobe-cms/worldwide-content/cdp/cf/ul/g/ef/3a/KNN.png`. Accessed: 2025-02-09.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

Gal Kaplun, Andrey Gurevich, Tal Swisa, Mazor David, Shai Shalev-Shwartz, and Eran Malach. Subtuning: Efficient finetuning for multi-task learning. *arXiv preprint arXiv:2302.06354*, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014. URL `https://arxiv.org/abs/1412.6980`.

Niels P. Kristensen, Malcolm J. Scoble, and Ole Karsholt. Lepidoptera phylogeny and systematics: the state of inventorying moth and butterfly diversity. *Zenodo (CERN European Organization for Nuclear Research)*, 12 2007. doi: 10.5281/zenodo.274044. URL `https://zenodo.org/record/274044`.

Takio Kurita. *Principal Component Analysis (PCA)*. 1 2021. doi: 10.1007/978-3-030-63416-2\{_}649. URL `https://doi.org/10.1007/978-3-030-63416-2_649`.

Eva J. Lewandowski and Karen S. Oberhauser. Butterfly Citizen Science Projects Support Conservation Activities among their Volunteers. *Citizen Science Theory and Practice*, 1(1):6, 5 2016. doi: 10.5334/cstp.10. URL `https://doi.org/10.5334/cstp.10`.

META. DINOv2: State-of-the-art computer vision models with self-supervised learning, 2023. URL `https://ai.meta.com/blog/dino-v2-computer-vision-self-supervised-learning/`. Accessed: 2025-02-03.

Andreas C Müller and Sarah Guido. *Introduction to machine learning with Python: a guide for data scientists.* " O'Reilly Media, Inc.", 2016.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. 2 2024. URL `https://arxiv.org/abs/2304.07193v2`.

Rita Pucci, Vincent J Kalkman, and Dan Stowell. Performance of computer vision algorithms for fine-grained classification using crowdsourced insect images. *arXiv preprint arXiv:2404.03474*, 2024.

PyTorch. ExponentialLR — PyTorch 2.6 documentation, 2025a. URL `https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ExponentialLR.html`. Accessed: 2025-02-11.

PyTorch. ReduceLROnPlateau — PyTorch 2.6 documentation, 2025b. URL `https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html`. Accessed: 2025-02-11.

PyTorch. StepLR — PyTorch 2.6 documentation, 2025c. URL `https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.StepLR.html`. Accessed: 2025-02-11.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019.

Zitao Shen. 3 min of Machine Learning: Cross Vaildation, 2020. URL `https://zitaoshen.rbind.io/project/machine_learning/machine-learning-101-cross-vaildation/featured_hubbc3c90efda00133c9815a673162bf54_130575_720x0_resize_lanczos_2.png`. Accessed: 2025-02-15.

University Stanford. Unsupervised Feature Learning and Deep Learning Tutorial, 2025. URL `http://deeplearning.stanford.edu/tutorial/supervised/SoftmaxRegression/`. Accessed: 2025-02-10.

Roger Vila. Importance of Lepidoptera — Proyecto Mariposa, 2025. URL `https://www.biologiaevolutiva.org/rvila/proyecto_mariposa/en/importance-lepidoptera`. Accessed: 2025-01-30.

Lingfeng Yang, Xiang Li, Renjie Song, Borui Zhao, Juntian Tao, Shihao Zhou, Jiajun Liang, and Jian Yang. Dynamic mlp for fine-grained image classification by leveraging geographical and temporal information. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10945–10954, 2022.

Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

Place, Date	Signature